

Jan 2018 Updated 24-1-18

IMPORTANT, the December 2017 update to the Pi Raspbian s/w required an update to one of the underlying TF libraries, and a significant recode of the main TF python s/w. To move to version rev 5 from an earlier version of the TF software you will need to do a clean install.

Note : The rev 5 s/w, now supports Dutch and French wordclock variants

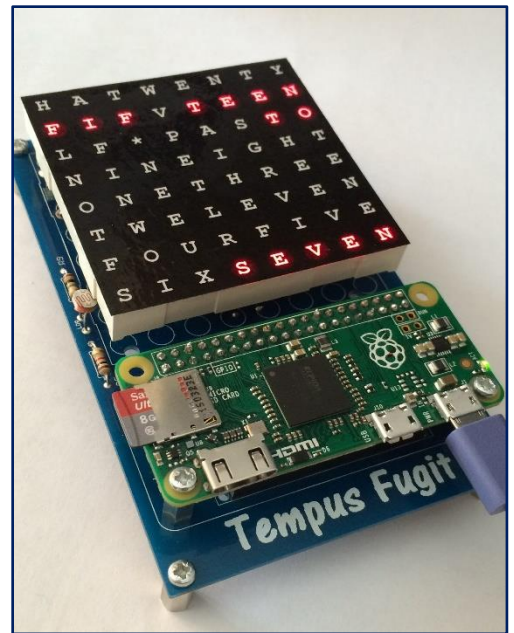
With the software fork to rev 5, the rev 3 is no longer supported, but remains available via my github site. [TFWordclock_Legacy folder]

The Tempus Fugit WordClock

The Tempus Fugit is a simple 8 by 8 character WordClock designed to work with either a Raspberry PiZero (original & W) or Arduino Nano board. The TF displays the time by forming words out of the 64 letters on the clock face, updated in five minute increments.

The TF WordClock is designed to be operated as a standalone board and includes a battery backed real time clock circuit to maintain the correct time when power to the main board is disconnected.

In addition to English the s/w also support time displays in French, Dutch or Latin, see section 9 for more information on this.



1 Introduction – How to use this document

This document contains all the information you should need to build, setup, operate and hack your TF WordClock. For those with ready built boards, who just want to get something working quickly you can jump to the section 3 ‘quick start’ and section 7 ‘Operating Instructions’.

The document is laid out in what you will hopefully find is a logical order starting with build instructions. The build is straightforward but there is the potential to solder parts in the wrong place so please don’t dive straight in with your soldering iron without at least glancing at the instructions first !

Where practical the s/w has been designed to make operating the TF WordClock the same for Raspberry Pi or NANO, where there are differences this is made clear in the instructions.

If you have opted to download and install software from GitHub, these instructions assume you are comfortable with basic operation of the Raspberry Pi / Nano and the Raspbian OS / Arduino IDE. If you are new to the Raspberry Pi / Arduino Nano you should get yourself familiar with their operation before attempting to setup your TF WordClock. There are lots of beginner’s guides on the internet;

- Raspberry Pi foundation material can be found at www.raspberrypi.org/help/
- Arduino can be found on their official website at www.arduino.cc/

If you are considering the TF WordClock for educational purposes, lesson plans and additional activity support material is available separately, these can also be adapted to suit specific needs if required.

1.1 Do’s and Don’ts

Rev 7 This document Copyright David Saul 2018

- is not a toy and is intended for use by or under the supervision of adults
- includes a small parts, keep out of the reach of small children



The TF WordClock connects directly to the Raspberry Pi's GPIO / Nano. Shorting these lines or applying voltages outside the specified limits could result in damage to the Raspberry Pi / Nano.

TF WordClock is intended as an electronics / software development project, in connecting your TF WordClock pledge reward to a Raspberry Pi or Arduino Nano computer you are accepting that the supplier / designer of the TF WordClock cannot be held liable for any consequential damage or losses how so ever caused.

The TF WordClock is designed to work with either the Raspberry Pi Zero or Arduino Nano. It has additionally been tested with Pi A+, B,B+ and Pi B Model 2 and 3. Mounting holes are included to all non PiZero variants to be fitted the rear of the TF WordClock board.

NEVER ATTEMPT TO OPERATE THE TF WORDCLOCK WITH A RASPBERRY PI AND NANO CONNECTED AT THE SAME TIME, IT IS VERY LIKELY THIS WILL DAMAGE ONE OR BOTH COMPUTER BOARDS

1.2 Revision History

Version	Key Changes
1	<ul style="list-style-type: none">• First formal issue
1b	<ul style="list-style-type: none">• Minor typo fixes and added missing info of display orientation on board
2	<ul style="list-style-type: none">• Updated to reflect the Python Rev 2 s/w update [s/w display rotation option added]• Improved consistency of terminology in build instructions• Added instructions for headless setup• <i>Table of content reformatted & separate index added</i> <p><i>Thanks to Simon Reap for updated 'rotation' code and headless instructions</i></p>
3	<ul style="list-style-type: none">• Updated to reflect the Python Rev 3 s/w update [s/w 'blink' options added]• Correct a couple of error in the instructions highlighted by users• Added some more information fixing problems with intermittent displays <p><i>Thanks to Ton Van Overbeek for updated for minutes flashing code</i></p>
4	<ul style="list-style-type: none">• Updated instructions for installing max7219 driver <p>Note:- the author of the max7219 python driver used in the TF wordclock development issued a major upgrade in Jan 2017, unfortunately there is no direct upgrade path from the old to the new version. The depreciated version is still available luckily but does require a change to down load and test instructions</p> <p>I will look at moving to the updated library but this could take sometime</p>
5	<ul style="list-style-type: none">• Minor update to cover new Pi Zero W
6	<ul style="list-style-type: none">• Correct a minor Typo and added additional instruction on non-English setups
7	<ul style="list-style-type: none">• Added info on compatibility issues with 29-11-17 Raspbian version
8	<ul style="list-style-type: none">• Rev 5 S/W update – move to luma.led_matrix – MAJOR UPDATE
9	<ul style="list-style-type: none">• French and Dutch variants now implemented in updated s/w plus some typo corrections



'Raspberry Pi' is a trademark of the Raspberry Pi Foundation



Arduino references are covered under a Creative Commons Attribution ShareAlike 3.0. licence

2 Contents

1	Introduction – How to use this document	2
2	Contents	4
3	Quick start	5
4	Build Instructions	7
5	Raspberry Pi Setup	12
6	Arduino Nano Setup	19
7	Operating instructions	20
8	Software Overview	23
9	Using the TF WordClock in other Languages	26
10	Fault finding	27
11	Appendix, Technical Information	30
12	Index	33

3 Quick start

Ok I know you just want to see your TF WordClock working but, before we start please to take a moment to read through the do's & don'ts on page 1.

This section assumes you have a built TF WordClock board together with a pre-programmed SD card or Nano and simply takes you through the basics of how to get your TF WordClock working

3.1 Connecting Pi / Nano to your TF WordClock

The PiZero / Nano are mounted to the top of side of the PCB. [Figure 1](#) and [Figure 2](#) show how the PiZero / Nano must be oriented, in their respective sockets.

For instructions on connecting other pi models see appendix 11.4

3.2 Fitting the RTC battery

The RTC battery is not supplied with any of the pledge rewards because of postage service restrictions. The battery type you need to purchase is CR1220, which is a 12.5mm diameter 3v lithium cell.

Note :- There are a number similar sized button cells based on other battery technologies, these only provide about 1.5V and will **NOT** work with the DS1307 RTC used on the TF Wordclock. If in doubt measure the voltage across the cell before you try to fit it.

Figure 3 Battery orientation



Figure 3 shows the correct orientation of the battery. If you have made you own TF Wordclock from a kit make sure you 'tinned' the battery pad correctly – see build step 3. This ensures RTC battery makes a good electrical connection.

Some online instructions for setting up the DS1307 RTC state that it will only work reliably with a backup battery fitted. Clearly it will not maintain the correct time with the power disconnected !, but I have not seen this as an issue in testing the prototypes boards.

Figure 1 PiZero orientation

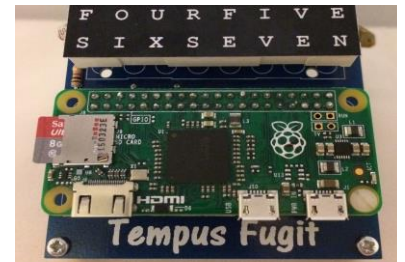
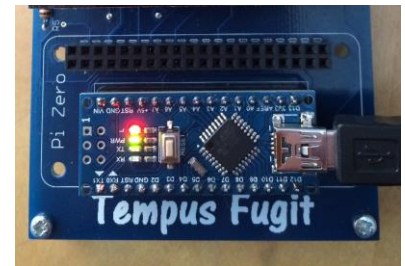


Figure 2 Nano orientation



3.3 Powering your TF WordClock

There are a variety of ways you can get power to the PiZero / Nano. The recommended way is to use a power supply conforming to the USB voltage standard [5V] with power connected via the Pi / Nano board USB connector. Note: connecting leads are not supplied with any of the backer pledge rewards.

If you want to power in any other way, please consult the relevant instructions from the [Raspberry Pi foundation](#) or [Arduino](#) website. The Tempus Fugit WordClock board derives its power from the Zero / Nano, it is NOT recommended to power it separately.

PiZero , all Raspberry Pi's use a micro USB connector for power – see Figure 1

- On the Zero this is the right hand of the two micro USB (viewed from above with GPIO upper most)
- Ideally you should use a PSU recommended for Raspberry Pi's, but The Zero and Tempus Fugit WordClock in combination draw a light load and I have found they work fine with an old blackberry phone charger
- For demos I have also used USB 'power banks' without a problem (see Nano note below)

Nano, The Nano clone uses a mini USB type USB connector – see Figure 2

- The Nano uses even less power than the Pi so any mini USB power adaptor should be fine, or lead connected to a PC etc.
- One issue I have seen when powering the Tempus Fugit WordClock from some USB 'power banks' is that the power draw is so low that they auto power off, thinking nothing is connected.

4 Build Instructions

This section details the relevant build processes for each of the TF WordClock variants, before starting do read the preparation section below.

4.1 Ready Built

Start from step 9, [only the stand-off pillars and Pi / Nano computer module need to be fitted].


4.2 Kit Built

Follow step 1 onwards.

4.3 Bare PCB

Follow step 1 onwards. A full parts list can be found in the appendix [11.2], to help you source the correct parts.

4.4 Individual build Steps

	<p>Preparation</p> <p>Please read through all the instructions before you start. The instructions assume a basic level of soldering competence, if you are in any doubt there is lots of material on web to help you – one example on the Adafruit site is - adafruit-guide-excellent-soldering</p> <ul style="list-style-type: none">• There are no large areas of copper, so a 25W or less soldering iron will be fine• When handling the ICs ensure you take static precautions• Make sure you fit the parts to the correct side of the PCB, the part is always fitted to the same side as it's silk screen decal• Be careful to fit the resistors in correct place• Make sure you get the polarised capacitors the correct way around in the correct places – it does matter• You don't have to use the IC socket supplied but it is recommended• If you need to force anything you are trying to put it in the wrong place!
1	<p>Parts Check</p> <p>Lay all the parts out to confirm you have everything. Use the part list in 11.2 to confirm nothing is missing</p> 

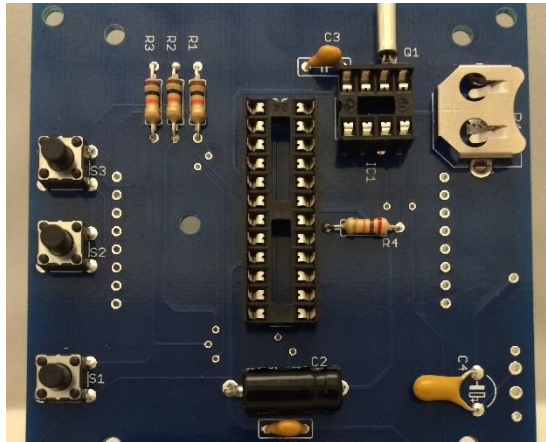
<p>2</p>	<p>Resistors</p> <p>Fit and solder resistors to the rear of the PCB;</p> <div data-bbox="186 315 305 504"> </div> <div data-bbox="354 325 592 493"> <p>10k – R1, R2, R3, R6</p> <p>4k7 – R7, R8</p> <p>27K – R4</p> </div>	
<p>3</p>	<p>Battery Clip</p> <p>Fitting this part is a two stage process, firstly tin the pad as shown in the figure - this will form the negative terminal for the battery. Next fit and solder the clip into position being careful to use as little solder as possible – see following note.</p> <p>Note :- Take particular care when soldering the battery clip to minimise any solder beads on the inside edges of the clip. This is to reduce the risk of the battery shorting out.</p>	<div data-bbox="722 682 1425 903"> </div> <p>Tin with solder here to create a very slightly raised pad area</p> <div data-bbox="743 997 1084 1129"> </div> <p>Avoid solder breakthrough here</p>
<p>4</p>	<p>Capacitors and Crystal</p> <p>Fit then solder the capacitors and crystal as shown. The electrolytic and tantalum capacitors are polarised and must be correctly oriented - as indicated on the silk screen. Note: the marked lead on the tantalum is the positive.</p> <p>Be careful to minimise repeat bending of the crystal leads.</p>	

5

IC Sockets and pushes

Fit then solder the 8 pin and 24 pin IC sockets together with the 3 off pushes S1,2,3 - making sure they sit flat to the PCB.

This completes the components which are fitted the Bottom of the PCB.



6

Connectors

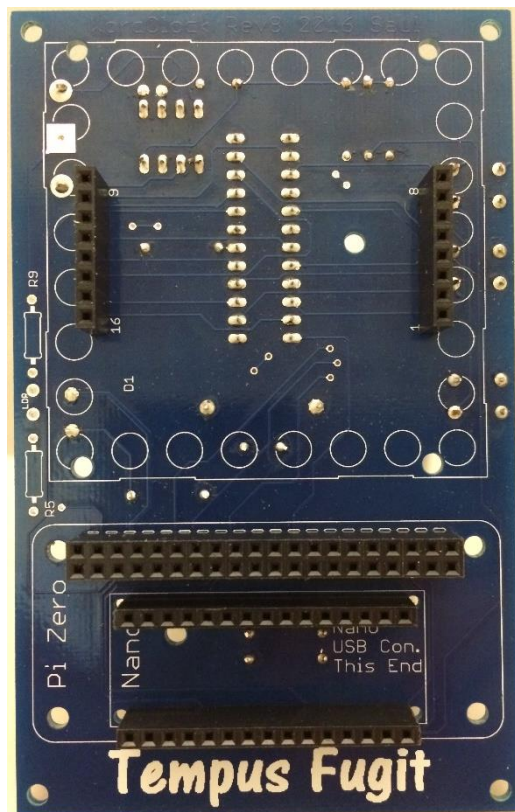
**** Be careful not to **
** accidentally fit these parts **
** to the wrong side of the **
** PCB ****




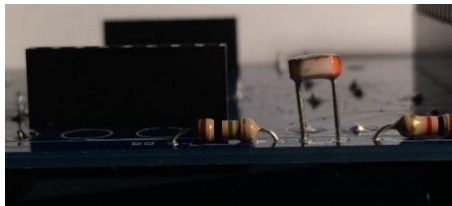
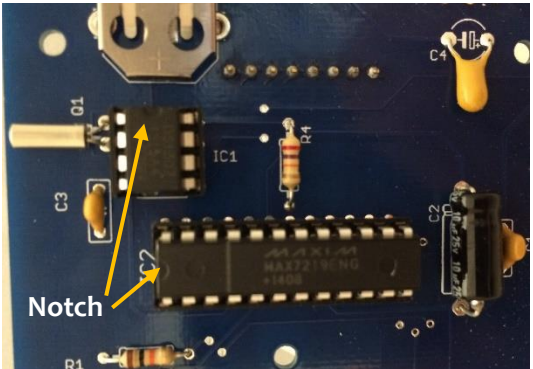
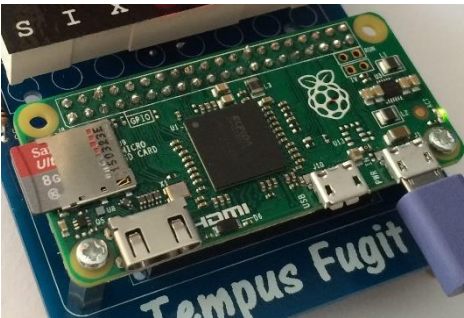
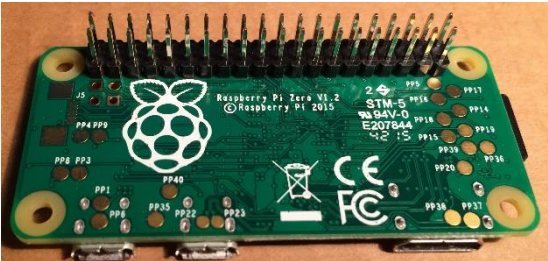

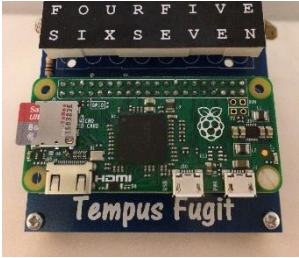
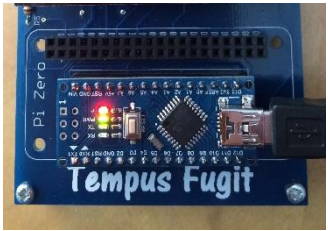
Fit then solder to the top of the PCB,



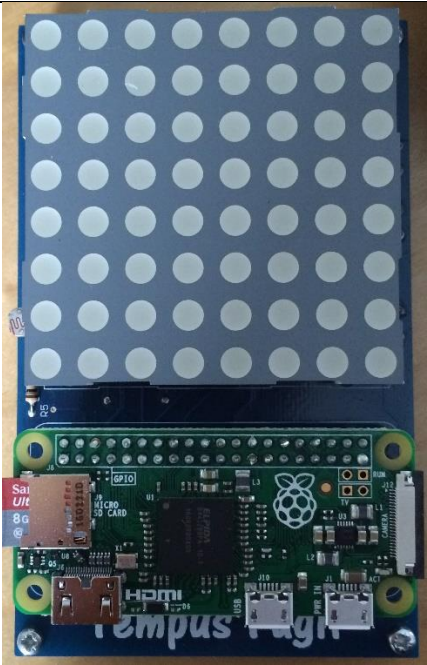
- 1 x 40 way Pi GPIO connector
- 2 x 18 way single in line Nano sockets
- 2 x 8 way single in line display sockets

To ensure the Pi / Nano boards and display are 'flat' to the PCB it is important to make certain the sockets are fitted square to the PCB. To do this firstly make sure the PCB is supported then solder a single corner pin first, check the alignment carefully re-soldering if needed. Next solder the opposite corner pin and again adjust the alignment by re-melting the solder if needed. Once you are happy that it is square solder the remaining pins.

**** Be careful not to **
** accidentally fit these parts **
** to the wrong side of the **
** PCB ****



<p>7</p>	<p>Top mounted resistors and LDR</p> <p>Fit then solder the final 2 remaining resistors and LDR to the top of the board.</p> <div data-bbox="186 321 410 457">  1k – R5  1M – R9 </div> <p>It is recommended to position the LDR so it sits below the display, if you fit it hard to the board take care to minimise the heat going into the LDR</p>	 
<p>8</p>	<p>IC's</p> <p>Ensuring you take precautions to discharge any static electricity carefully fit and solder the 2 IC's checking they are the correct oriented.</p>	
<p>9</p>	<p>Display and Pi / Nano</p> <p>For the PiZero you will have to fit and solder a 40 [20+20] way connector to the GPIO pads first, this is included in the kit and 'PiReady' options</p> <p><u>Note this has to be fitted so it is downward facing – see the diagram</u></p> <p>Fit either a PiZero or Nano in their respective sockets, as shown in figure 1 and 2.</p> <p>The PiZero should be supported using 2 hex stand-offs [supplied].</p> <div data-bbox="256 1549 717 1864">  </div>	<div data-bbox="844 1018 1388 1276">  </div> <div data-bbox="844 1287 1388 1549">  </div> <div data-bbox="815 1575 1112 1831">  </div> <div data-bbox="1123 1602 1448 1831">  </div>

10	<p>Battery</p> <p>Fit the RTC battery making sure it is the correct way around as pictured.</p>	
11	<p>Fitting the Display</p> <p>Finally fit the display as shown. To orientate the display correctly make sure the supplier id text is on the right of the display as the viewed from above – see diagram below;</p>  <p>It is much easier to align the letter template with the display fitted so don't be tempted to do this yet, see section 11.5 for more on this.</p>	 <p>← manufactures text on display</p>
12	<p>Mechanical fixing</p> <p>To complete the build fit a stand-off to each corner of the board, using m2,5 pan head screws supplied.</p>	
13	<p>Visual check</p> <p>Your Tempus Fugit WordClock is now complete and ready to test, before you connect power visually check the completed board against the part list shown section 11.2, to make everything is fitted in the correct place. As you do this also look for any dry joints or no solder shorts.</p>	

5 Raspberry Pi Setup

These instructions are written for rev 5 onwards s/w for rev 3 please refer to version 7 of this document . This and the rev 3 code can be found in the TFWordclock_legacy GitHub folder.

The most recent s/w release these instructions have been tested against is the Raspbian release 2017-11-29- Kernel Version 4.9

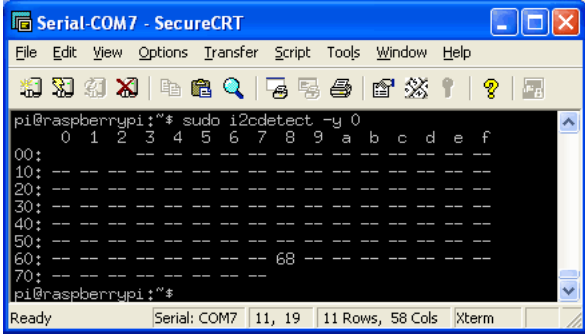
5.1 Assumptions

These instructions assume;

- You are installing version 5 or later TF s/w
- Your TF wordclock is built and connected to a raspberry Pi as shown in 3.1
- You have fitted a battery for the RTC chip as shown in 3.2
- Your pi is connected to the internet
- You are using a November 29th 2017 or later Raspbian Jessie version and a B+ or newer Rpi board, created from either Noobs or the dedicated Raspbian image downloads. If you are using the 'Jessie-lite install', you will have to "apt-get install git" to allow the git commands.
- You are working from a 'clean' Raspbian image, if you have previously configured other i2c hardware on the SD card some instructions may return an error / notification response saying 'xxx' is already installed / the latest version. This is fine just move on to the next instruction.

5.2 Configuring the RTC [I2C interface]

From the console or a *terminal* window in the graphical user interface [GUI] enter the following commands [note they are case sensitive].

Action	Command(s) to enter
1 Ensure you are running latest s/w	<code>sudo apt-get update</code> <code>sudo apt-get upgrade</code>
2 Enable I2C using rasp-config	<i>Via on screen system menu select- preferences / Raspberry Pi Configuration / Interfaces</i>
3 Reboot	<code>sudo reboot</code>
4 Install i2c-tools	<code>sudo apt-get install python-smbus i2c-tools -y</code>
5 Check RTC i2c interface is working, you should see;	<code>sudo i2cdetect -y 1</code> 
6 Setup DS1307 interface 1 of 3	<code>sudo bash</code>
7 Setup DS1307 interface 2 of 3	<code>echo ds1307 0x68 > /sys/class/i2c-adapter/i2c-1/new_device</code>
8 Setup DS1307 interface 3 of 3	<code>exit</code>
9 Confirm hwclock is working ok The system should respond with day and time	<code>sudo hwclock -r</code>
10 Check you can write to the RTC If it works you will NOT see any response message	<code>sudo hwclock -w</code>

Action	Command(s) to enter
<p>11 Set it up to auto run the ds1307 application. To do this you need to edit rc.local, it should look like this;</p>  <pre> GNU nano 2.2.6 File: /etc/rc.local #!/bin/sh -e # # rc.local # # This script is executed at the end of each multiuser runlevel. # Make sure that the script will "exit 0" on success or any other # value on error. # # In order to enable or disable this script just change the execution # bits. # # By default this script does nothing. # # Print the IP address _IP=\$(hostname -I) true if ["\$_IP"]; then printf "My IP address is %s\n" "\$_IP" fi echo ds1307 0x68 >/sys/class/i2c-adapter/i2c-1/new_device exit 0 </pre>	<p>sudo nano /etc/rc.local</p> <p>add the following line before 'exit o'</p> <p>echo ds1307 0x68 >/sys/class/i2c-adapter/i2c-1/new_device</p> <p>control x and Y to save and exit nano</p>
<p>12 Reboot to check you have made the modification correctly. The system should respond with day and time</p>	<p>sudo reboot</p> <p>[then when it has rebooted] sudo hwclock -r</p>
<p>13 Lastly we need to edit hwclock-set to make sure the system time does not automatically overwrite the hwclock time when the Pi boots. This is important if you are operating your TF Wordclock standalone without a internet connection, it should look like this;</p>  <pre> GNU nano 2.2.6 File: hwclock-set #!/bin/sh # Reset the System Clock to UTC if the hardware clock from which it # was copied by the kernel was in localtime. dev=\$1 # # if [-e /run/systemd/system] ; then # exit 0 # fi # # if [-e /run/udev/hwclock-set] ; then # exit 0 # fi # # if [-f /etc/default/rcS] ; then # . /etc/default/rcS # fi # # These defaults are user-overridable in /etc/default/hwclock BADYEAR=no HWCLOCKACCESS=yes HWCLOCKPARS= HCTOSYS_DEVICE=rtc0 if [-f /etc/default/hwclock] ; then . /etc/default/hwclock fi if [yes = "\$BADYEAR"] ; then </pre>	<p>cd /lib/udev</p> <p>sudo nano hwclock-set</p> <p>comment out;</p> <p>if [-e /run/systemd/system] ; then</p> <p>exit 0</p> <p>fi</p> <p>control x and then y to save & exit nano</p> <p>then reboot</p> <p>sudo reboot</p>

Table 1, RTC i2c interface setup

5.3 Max7219 [SPI interface]

The following instructions explain how to install.

There are a number of other python libraries for driving the Max chip but the TF WordClock software will only work with this library. If you want to use another library, you will need modify the TF WorkClock s/w.

From the console or a *terminal* window in the graph user interface [GUI] enter the following commands - note they are case sensitive.

Action	Command(s) to enter
1 Ensure you are running latest s/w	<code>sudo apt-get update</code> <code>sudo apt-get upgrade</code>
2 Enable SPI using rasp-config	<i>Via on screen system menu select- preferences / Raspberry Pi Configuration / Interfaces</i>
3 Reboot	<code>sudo reboot</code>
Confirm SPI is enabled You should see something like this <pre>crw----- 1 root root 153, 0 Jan 1 1970 /dev/spidev0.0 crw----- 1 root root 153, 1 Jan 1 1970 /dev/spidev0.1</pre>	<code>ls -l /dev/spi*</code>
4 Install dependencies	<code>sudo usermod -a -G spi,gpio pi</code> <code>sudo apt-get install build-essential python-dev python-pip</code> <code>sudo apt-get install libfreetype6-dev libjpeg-dev</code>
5 Down load the luma.led_matrix library from github This may well take a while to complete and you could well get a number of warnings but it should complete with 'successfully installed luma.led_matrix'	<code>sudo -H pip install --upgrade luma.led_matrix</code> <code>cd [return to pi@raspberrypi:~ \$ prompt]</code>

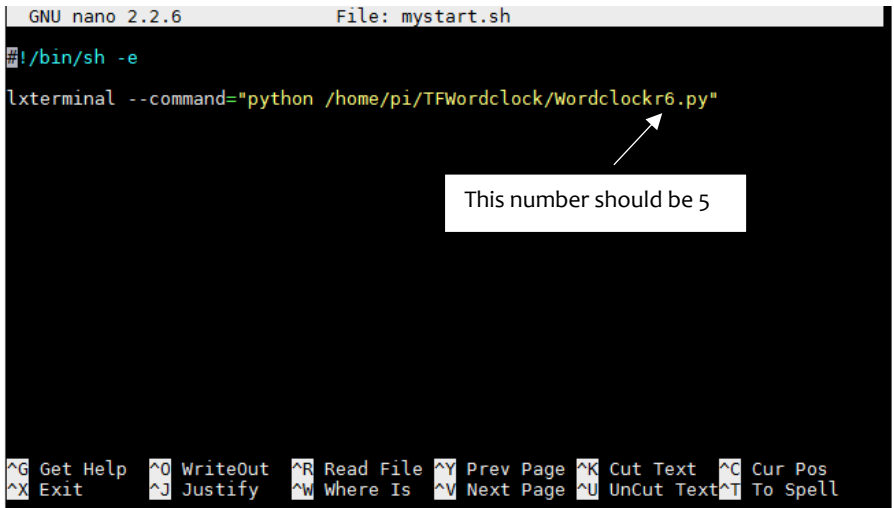
Table 2 SPI setup

5.4 Installing the TF WordClock Python s/w

The following instructions will download the TF WordClock s/w and take you through setting the s/w to start automatically when the pi boots up.

These instructions assume you boot you pi directly to the GUI [desktop].

From the console or a *terminal* window in the GUI enter the following commands [note they are case sensitive].

Action	Command(s) to enter
1 Ensure you are running latest s/w	<code>sudo apt-get update</code> <code>sudo apt-get upgrade</code>
2 Download the TF WordClock software from github	<code>git clone https://github.com/DavidMS51/TFWordclock.git</code>
3 Testing library installation	<code>cd TFWordclock</code> <code>python pixtest1.py</code> [You should see a series of flashing letters on the TF wordclock display] <i>Terminate with control z</i>
4 Setup s/w to auto execute when Pi boots a. Create a file called 'mystart.sh' on your desktop	<code>cd /home/pi/Desktop/</code> <code>nano mystart.sh</code> enter / copy the following line into mystart.sh – these need to be entered exactly as shown. <code>#!/bin/sh -e</code> <code>lxterminal --command="python /home/pi/TFWordclock/Wordclockr5.py"</code> check it looks like the screen shot below, then save and exit nano with control x and y. 
5 b. Make 'mystart.sh' executable	<code>chmod +x mystart.sh</code>

Action	Command(s) to enter
<p>3 c. Make an autostart file to run 'mystart'</p>	<p>From pi@raspberrypi</p> <p><code>mkdir /home/pi/.config/autostart</code> [don't worry if you get a 'file exists error']</p> <p><code>nano /home/pi/.config/autostart/auto.desktop</code></p> <p>Enter / copy the following line into mystart.sh – these need to be entered exactly as shown.</p> <p>[Desktop Entry]</p> <p>Type=Application</p> <p>Exec=/home/pi/Desktop/mystart.sh</p> <p>check it looks like the screen shot below, then save and exit nano with control x and y.</p>  <p>The advantage to this 2 stage approach is that you can more easily edit and test you autorun file than if you tried to include everything in the single auto.desktop script</p>
<p>4 It is worth making sure the hwclock time is correct while you are connected to the internet. You can set the time using the TF's pushes but not the date – which means winter daylight saving time may switch over on the wrong day !</p>	<p>Check time is correct on Pi</p> <p>Force update from internet - <code>sudo timedatectl</code></p> <p>Force RTC clock on TF to update - <code>sudo hwclock -w</code></p>
<p>Your Done, if you reboot now your TF Wordclock should start automatically, in English. If you are wanting to run it in one of the other supported languages you will need to modify the 'mystart.sh' file - see section 9</p>	

Table 3 hwclock setup

5.5 Non-standard / Alternative setups

The following bullets are included to provide limited support for people looking to use TF Wordclock for a 'non-standard' Pi setup – ie as described in 5.1.

For more / the latest information on non-standard setups please check out the [TF Google Community](#).

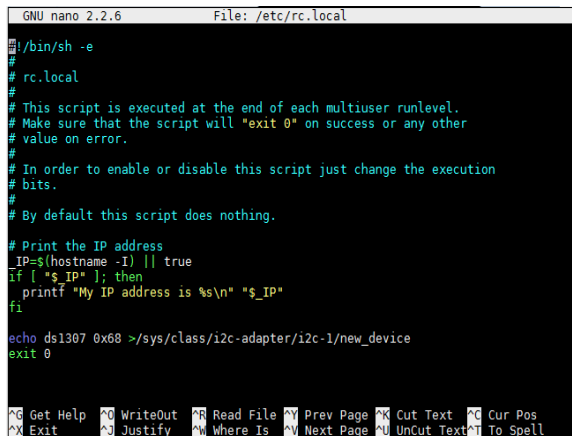
- If you are using the 'Jessie-lite install', you will have run "*apt-get install git*" to allow the git commands to work as described
- If you are not using the GUI environment you will need to use "*sudo raspi-config*" to enable the I2C and SPI interfaces
- The following instructions provide an alternative auto start option – again aimed primarily at people not wanting to boot into the GUI

In addition to instruction 11 in 4.2 [shown below] add the follow instruction following the *echo ds1307* line.

```
su - pi -c "python /home/pi/TFWordclock/Wordclockr3.py &"
```

In this case you should then skip section 4.4 step 3 onwards

- 11 Set it up to auto run the ds1307 application. To do this you need to edit rc.local, it should look like this;



```
GNU nano 2.2.6 File: /etc/rc.local
#!/bin/sh -e
#
# rc.local
#
# This script is executed at the end of each multiuser runlevel.
# Make sure that the script will "exit 0" on success or any other
# value on error.
#
# In order to enable or disable this script just change the execution
# bits.
#
# By default this script does nothing.
#
# Print the IP address
IP=$(hostname -I) || true
if [ "$IP" ]; then
  printf "My IP address is %s\n" "$IP"
fi
echo ds1307 0x68 > /sys/class/i2c-adapter/i2c-1/new_device
exit 0
```

sudo nano /etc/rc.local

add the following line before 'exit 0'

echo ds1307 0x68 > /sys/class/i2c-adapter/i2c-1/new_device

control x and Y to save and exit nano

6 Arduino Nano Setup

If your pledge reward includes a Nano it will come preloaded with the current Tempus Fugit WordClock application. Just plug it in and connect power.

These instructions assume the reader is familiar with the Arduino IDE. They have been tested in the main with the IDE running on a PC under windows 8.1 and 10, and in a more limited way with the latest IDE running on a Pi3 - see my blog for details on how to set this up [don't use apt-get as this installs a very old version of the IDE].

To work on the Arduino TF WordClock source code you will need to install the following libraries, in the normal manner from zip files.

RTC interface - LedControl

Display driver – RTCLib-master

All the development work on the Arduino code was done with the default settings on the IDE so you should simply have to select Nano as the target board. I would however make sure everything is working correctly by compiling the classic Arduino code to flash the on board LED before you try doing anything with the Tempus Fugit WordClock source code.

The Source for the language variants is different, see section 9 for details on this.

7 Operating instructions

Operating instructions for the Pi and Nano versions of the WordClock are generally the same. These instructions assume you have;

- the relevant s/w installed and setup to auto start at power up
- installed the RTC backup battery
- have fitted the English word template and correctly aligned it

Once the time has been set as part of the TF WordClock should initialise automatically to the correct time, at power-up.

An alignment mode is available to assist with fitting the word template correctly, this is described in section 7.3.2.

All functions are accessed by momentary switches S1, S2 and S3.

7.1 First Operation

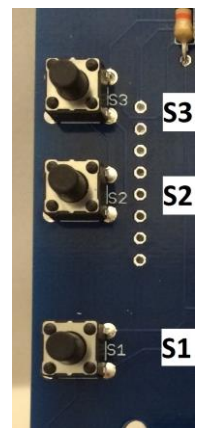
Before you power up your TF WordClock for the first time make sure the battery is correctly fitted. If you opted for the kit reward you will also need to temporarily attach a word template. Full instructions / tips for aligning the word templates can be found in appendix 11.5.

You may see an error display [see 10] when you first apply power, this is most likely just indicating the that the RTC has not been initialised. If you see this on subsequent power-ups it indicates other potential problems refer to section 10 for help on fault finding.

7.2 Initialisation

When you apply power the Pi / Nano will initially go through and internal initialisation process before it is ready to run the dedicated TF WordClock application. For a PiZero this will typically take about 30 seconds for the Nano it is much quicker.

Figure 4 pushes



7.3 Starting the TF WordClock applications

The Nano version of the TF WORDCLOCK application will auto-start when powered up. With Raspberry Pi versions the pre-loaded SD cards are programmed also to auto start the TF WordClock s/w. see section 5.4 if you are setting up the RaspberryPi yourself.

If you want the TF WordClock s/w to start manually and have installed it as per the instructions in 5 enter the following into a terminal window or at the command line prompt;

`'python /TFWordclock/Wordclockr5.py'`

[note the '5' will change if the software is updated in the future]

For the full usage syntax see section 8.2.

The TF WordClock will run through 2 or 3 initialisation screens when the software starts,

- 'INIT' – Nano only, see forced RTC setting below
- 'V' and 'ONE', displays current software version
- A rolling set of columns [Nano] or countdown followed by INIT [Pi]

The TF WordClock will switch automatically to display the time display when the initialisation screens complete.

During the initialisation process there are 2 special functions you can access.

7.3.1 Forced RTC setting – Nano only

By holding down the 'up' [s3] and 'down' [s2] pushes together when the word 'INIT' is displayed you can force the Nano to reload the TF WordClock RTC with the date and time the s/w was last compiled. This is really just included for fault finding to allow the RTC to be forced to a known state. With the 2 pushes held down you will see a diagonal line slowly being displayed. If you release the pushed before it completes the RTC update will abort and initialisation will continue as normal. Otherwise it will update the RTC and initialisation will continue.

This feature is not included in the Pi version s/w as you can more effectively fault find from the RaspberryPi command line.

7.3.2 Jump to Demo / Align modes

If you momentarily hold down the select push [s1] during the rolling column / countdown display, the s/w will jump to a demo mode where it quickly run through hours, minutes and then a combined display. You can revert back to a normal clock display by momentarily holding down the select push [s1]. Note it is only possible to abort the demo mode once you are in the combined display loop.

Align mode, if rather than momentarily holding down select push [s1] during the rolling column / rows display you keep the push held down the display will switch to a static display to make aligning the word template easier. You can revert back to a normal demo mode by momentarily holding down the 'up' [s3] and 'down' [s2] pushes together.

7.4 Normal display mode

When the TF WordClock is in its normal display mode the activity Led [pi] or L Led [Nano] will flash every second. In this mode there are 3 functions you can access.

7.4.1 Setting the time

If you momentarily hold down the select push [s1] The display will be blanked and the ‘*’ will be illuminated.

- Momentarily holding down the up push [s3] will allows you to cycle through the hours display
- Momentarily holding down the down push [s2] will allows you to cycle through the minutes display

It is easiest to set the hours first then minutes

When you have the time correct hold [and keep] down the select push [s1], you will see a diagonal sequence extending across the display when this completed the RTC will be updated with the new time you have selected [on the pi ‘U’ will also flash twice]. If you release the select push before the diagonal line completes the update will be aborted.

If you leave the display in time set mode for more than 5 seconds without touching the pushes the s/w will time out and revert back to the normal clock display.

If you are running the TF WordClock with a raspberry Pi and can connect to a screen / keyboard it is better to set the RTC time via the operating system by first making sure the system time is current then performing a ‘sudo hwclock –w” command, this ensures the date is also correctly set.

7.4.2 Adjusting the display brightness

The TF WordClock automatically adjusts the LED display brightness depending on the ambient light level. Depending how opaque the word template is that you are using, you may want to reduce the maximum brightness. With the clock displaying the time normally you can do this by momentarily holding down the ‘down’ push [s3] to reduce the maximum brightness or the ‘up’ push [s2] to increase. The automatic level control will then work to the new high limit.

During the setting process the ‘*’ will flash each time you adjust the level and ‘H’ or ‘L’ will flash if you reach the maximum or minimum settings - there are 15 levels in total.

The new maximum level will be stored between power cycles.

Note: With the Nano there is up to a 2 minute delay between changing the setting and the new setting being written to the Nano’s EEPROM memory, if you power down before this the new value will be lost on. With the Pi the updated value is stored as part of the power-down process.

7.4.3 Power down – Raspberry Pi only

To avoid the risk of SD card corruption it is important you power the Pi down correctly. To do this with TF WordClock hold and keep held down the up [s3] & down [s2] pushes, the TF WordClock will count down from nine to one, when this completes the display will blank and the pi will start to shut

down. The activity LED on the Zero board will flash a number of times then stay on to indicate it is ok to remove the power.

For the Nano version it is fine just to remove the power.

8 Software Overview

The TF WordClock s/w is probably best described as functional, I have tried to divide it up into sensible blocks but it does suffer from me adding extra feature as I went along. I have tried to make sure it is well if not fully commented, but if you want some more information on any specific elements please do get into contact.

With the Python code I have tried to avoid dependencies where possible to minimise the risk that unrelated changes to the RaspberryPi operating system upsets the code. With the Nano running compiled code this only becomes an issue if you re-compile and download new code from the IDE.

Although there is no need to connect a monitor both versions do provide quite a lot of feedback if you have a screen connected – or the IDE with serial monitor running for the Nano.

8.1 Nano

The Nano is structured in the classic Arduino manner in 2 parts, the first a single shot execution, used to set things up initially, then the main code which runs cyclically. The latter is broken down into a number of functions. There are a couple of extra libraries that you will need to install into the IDE on your desktop if you want to modify and recompile the code. Where licencing allows me to distribute them they will be included in the main github for the WordClock, otherwise details will be provided of where they can be downloaded from.

With the Nano rather than over complicate things I have opted to go for completely different applications for each language. All pre-loaded Nano clones will run the English version. See section 9 for more details on how to set the Tempus Fugit WordClock up for other languages.

The Nano s/w does not currently support display rotation.

If you want to hack the code to create your own language version, the core time display code is in the *'displayTime'* function, with a few minor code elements providing key press feedback spread around other places [you can most likely leave these unchanged].

8.3 Raspberry Pi – Python

The first important point to note is that the Python code is written to run on version 2.7.x, it will NOT run on 3.x. If at some stage the libraries used are all updated to 3.x it should be relatively easy to produce a version what will work on Python 3 but it is not a priority currently.

The Python code is in 2 parts,

The core application ‘TFWordclockr*x*.py’ [where *x* is the version number, currently 5]

- This contains the core functionality for timing, and extra functionality such as setting the time. To display the words routines in the display subroutine file are called.

The display subroutine file ‘timewrd*x*ca_eng.py’ [where *x* is the version number, currently 5]

- This contains a series of subroutines to display the individual words and related functions
- The French variant is called ‘timewrd*x*ca_fr.py’ and Dutch ‘timewrd*x*ca_du.py’]

These files need to be in the same folder [home/pi/TFWordclock is the default].

By setting up the code in this way it is possible to add new language versions with minimal change to the core application code – you just point at a different version of timewrd3ca_xx.py. It is also means that you can change the display type without changing the core code as at the display function, The current distribution supports English, French and Dutch directly.

The full usage syntax is:

usage: Wordclockr3.py [-h] [-b bklopt] [{English,French,Dutch,German}]

Tempus Fugit Word Clock

positional arguments:

{English,French,Dutch,German} [note German not currently supported]

Language: English, French, Dutch or German (default:English)

optional arguments:

-h, --help show this help message and exit

-b bklopt, --blink bklopt

blink - 0 = board act led only 1:1, 1 = minute indication blinking, 2 = board act led only 1:20,
3 =act led off (default: 0)

There is also a trial Latin version on GitHub, currently this uses separate code because the manner of telling the time in Latin is too different to that used in modern languages. For more on language options see section 9.

8.3.1 Display rotations,

From version 2 of the s/w you can rotate the display, this allows you to operate the TF WordClock to display words correctly with the board in a different orientation. This is illustrated in the picture below.

With rev5 the rotation is hardcoded into the timrwr5ca_eng.py file, rather than via the command line. Changing the rotation only require a single variable to be changed in the code

Towards the start of the code look for the following

```
#-----  
rotation = 0          #set global rotate (0,1,2,3)  
#-----  
serial = spi(port=0, device=0, gpio=noop())  
device = max7219(serial, rotate = rotation)
```

The options for are: 0 – no rotation, 1 - rotated 90 clockwise, 2 - upside down and 3 = 90 counter-clockwise

Set 'rotation' to the required setting and save the file and rerun the TFWordclock application.

Hopefully it is obvious that you also need to re-attach the letter template in the correct orientation to the LED matrix display.

8.3.2 Blink Options,

From version 3 of the s/w a number of 'blink' options have been added to the display. These are selected using the '-b x' suffice when starting the application;

- 1:1 board act led only [-b 0]
 - This original operating mode with the Pi's activity led flashing once a second
- minute indication blinking [-b 1]
 - In this mode the Pi's activity led is flashed briefly once every 60 seconds – so you know the Pi is working but it is less distracting
- board act led only 1:10 [-b 2]
 - In this mode the Pi's activity led is flashed briefly once every 10 seconds
- act led off [-b 3]
 - Here the Pi's activity led is completely of, great if flashing things annoy you but does mean you have to watch the clock for 3 minutes to know it is working!

For example – `Wordclock5.py -b 1` will flash the led once every minute for about 10th of a second

Irrespective of the 'blink' you select the activity led on the Zero will return to its normal mode indicating board activity as the Pi goes through the power down sequence, so you can still easily when it is safe to remove power.

You can combine the various start-up options for instance;

`Python Wordclock3.py Dutch -b 2`

will display in correctly with ha Dutch word template, with TF board on it's side [like the picture at the start of this section] and the activity led flashing briefly once every 10 seconds.



9 Using the TF WordClock in other Languages

As an alternative to English the TF WordClock can be setup to work in French, Dutch and Latin. The French and Dutch version have been briefly checked by native speakers but a lot less work has gone into them than the English version. In the case of Latin this has been coded based on information taken from the internet. The French and Dutch versions will automatically pick up any s/w updates as they share a common python code core with the English version. Unless there is specific interest in the Latin version it is not planned to work on this further.

9.1 Dutch

This has been coded for both the Pi and Nano. As with the English version time is divided in to 5 minute intervals.

To switch to Dutch on the Pi simply add the word 'Dutch' at the end of the command line

For example, python 'TFWordclockr2.py Dutch'

To set it to autostart in Dutch, comment out the English version in the mystart.sh file on the desktop and uncomment the Dutch version.

On the Nano you will need to connect the TF WordClock up to a computer running the Arduino IDE and the download the Dutch code version – this can be found in the Tempus Fugit WordClock GitHub repository. You may also need to download some additional libraries – see section 6.

9.2 French

This has only been coded for the Pi currently. Additionally, because of the limited number of characters combined with the way you tell the time in French the TF WordClock works in 15minute intervals rather than 5.

To switch to French on the Pi simply add the word 'French' at the end of the command line.

For example, python 'TFWordclockr2.py French'

To set it to autostart in French, edit the relevant command line to read as the example above in the mystart.sh file on the desktop.

9.3 Latin

This has only been coded for the Pi. The Latin version display hours during the day and four 3 hour slots during the night. As Latin does not have words for periods of less than an hour and overnight they were only interested in the changing of the guards. It is also worth noting that The hour as a period varied in length with the seasons, so they would always have 12 daylight hours- the software does not attempt to mimic this.

10 Fault finding

The ready built TF WordClock s are all tested before shipping and for the kits versions active components are functionally tested before being packed in anti-static bags, component faults are therefore unlikely. More likely they will be down to;

- Incorrectly assembled kits
- Parts that have come lose in the shipping process
- Incorrectly installed s/w – or a change to the software environment such as a Linux kernel update

The table below provide some suggestions based on problems I have seen in developing the prototypes and previous electronic designs. Although you can do some fault finding with the board standalone, you can get a lot better idea of what is happening [or not] by connecting up to a screen and keyboard.

Pi, connect up a monitor / keyboard / mouse as you would normally [you can find instructions the foundation website if needed] if possible directly – you will need a USB hub and adaptor cable for a PiZero. The Application produced a number status message as it starts and runs which should help with diagnosing problems

Nano, connect to a computer with the Arduino IDE loaded and connect to the board via a USB lead [see section 6]. You can use the serial monitor application [set baud rate to 9600] in the IDE to look at the status message the application produces as it starts and runs

Fault	Things to Look at
Display blank or random characters displayed	<p>Pi only - have you performed an apt-get update or equivalent after November 29th ?</p> <ul style="list-style-type: none"> perform clean install with Rev 5 and supporting libraries [preferred] perform clean install with rev 3 code and pre November 2017 Raspbian issue [non preferred] , DO NOT perform an apt-get upgrade
Display remains blank	<ul style="list-style-type: none"> Check the power is connected correctly <ul style="list-style-type: none"> Is the power Led on the Nano on? Activity Led on Pi Zero flashing On a Nano is the 'L' Led on or the 'activity' Led on the Pi flashing if yes this indicates the code is running correctly <ul style="list-style-type: none"> Check the Nano / Pi is fitted correctly Check the display is not just very dim If you built from a kit check all the components are correctly fitted and soldered Check the display is fitted correctly and has not become lose Connect up Nano / Pi to a host computer / screen & keyboard, to confirm the Nano / Pi are working [see start of section 10] <ul style="list-style-type: none"> Try a different SD card that you know works with a PiZero On the Nano try downloading a simple program to flash the on board Led
Incorrect characters displayed	<ul style="list-style-type: none"> Miss-match between letter template language and s/w language setting Miss-match between letter template orientation and s/w rotation setting

Continued on following page

Fault	Things to Look at
Correct Time is not maintained when powered down	<ul style="list-style-type: none"> • If built from a kit check parts are correctly fitted and you have followed the soldering instructions for the battery clip [see build instructions 3 and 10 in section 4.4] • Check you have the correct Battery type [CR 1220] • Check the battery voltage is between more than 2V with voltmeter [the easiest way to do this is between the outside of the battery clip (+) and the square pad on the top side of the PCB]. <ul style="list-style-type: none"> – If the voltage is low [and you know the battery is new] remove the battery and see if the voltage recovers, if so suspect a short between the terminals- see build step 3 in section 4.4 for more on this. – If the voltage is fluctuating suspect a bad battery connection, check the square pad which forms the negative connection is tinned to create a very slightly raised area, see build step 3 in section 4.4 for more on this • Connect up Nano / Pi to a host computer / screen & keyboard, to confirm [see start of section 10] this will allow you see / check the RTC is working <ul style="list-style-type: none"> – On a pi check you can read and write the hwclock [see 5.2], if not try removing all power from the board [in the backup battery] , leave for a few minutes then reconnect – On the Nano check the clock time is change on the serial monitor line, leave for a few minutes then reconnect
Rows or columns on the LED display remain blank or only light intermittently	<ul style="list-style-type: none"> • If built from a kit check parts are correctly fitted, particularly the MAX7219 / Pi / Nano in their sockets • Check the Led display module is correctly seated in the single in line sockets • If the problem is intermittent, <ul style="list-style-type: none"> – It may be down to a poor connection between the display and single inline sockets, if the problem goes away if you angle the display in the socket this is a good indication of the problem. You can often fix this by slightly angling display pins a few degrees off vertical using a wooden 'ice lolly stick' or similar to carefully bend all the pins on one side at the same time. – It could also be down to a bead of the plastic [used to encapsulate the device] on a pin. You can generally see this if you carefully remove the display and check each pin with a magnifying glass. Beads can be removed by gently scraping the back of a knife up the pin these should only be attempted by an adult and the relevant precautions for using an open bladed knife must be followed [see manufactures instructions for use]
The display is very dim	<ul style="list-style-type: none"> • Check the display brightness limit has not been set to a low value see section 7.4.2 • You are using a normal rather than 'High Red' Kingbright display • The optical sensor is being shaded by the board / enclosure or has got bent so it is under the display

Fault	Things to Look at
The display shows a flashing 'E E E' message [Pi only]	<ul style="list-style-type: none"> • One of more switch is being held down during the initialisation process. • The pushes have been incorrectly fitted or not fitted [This feature is included to avoid the risk that incorrectly fitted pushes s2 and s3 could be erroneously identified by the s/w as active causing the pi to shut down]. If the fault continues for 5 minutes the application will quit
Arduino IDE will not recognise the Nano Clone	<p>Most Nano clones including those supplied with the TF WordClock use CH340G USB driver chip to keep the part cost down.</p> <ul style="list-style-type: none"> • CH340G USB driver is installed on your MAC / PC, there are a number of sites on the internet where the driver can be down loaded for free. <p>Note: - I have not seen this problem with Windows 8.1 / 10 or the new Linux ARM (experimental) IDE, but generally with Windows7 you seem to need to install a CH340 driver</p>

11 Appendix, Technical Information

11.1 GPIO assignments

Pin name		Description
Pi	Nano	
GPIO 02 [3]	A4	i2c SDA
GPIO 03 [5]	A5	i2c SCL
GPIO 24 [18]	A7	LDR Sense
GPIO 23 [16]	D2	LDR source voltage
GPIO 22 [15]	D7	S3
GPIO 17 [11]	D8	S2
GPIO 27 [13]	D9	S1
GPIO 08 [24]	D10	SPI Load
GPIO 11 [23]	D11	SPI CLK
GPIO 10 [19]	D12	SPI Din

11.2 Parts List

Parts are available from a number of suppliers, although you may need to shop around for the Led display. Between them all the parts can be purchased from [Rapid](#), [Bitsbox](#), [CPC](#) and [Toby](#) in the UK

Ref	Part Description	Quantity	Notes
C1,C3	100nF	2	Be careful not to mix up C2 and C4
C2	10uF – Electrolytic	1	
C4	10uF- tantalum	1	
D1	TA23-11SRWA	1	
IC1	DS107	1	
IC2	MAX7219	1	
LDR2	LDR04	1	
Q1	32kHz watch crystal	1	
R1,R2,R3,R6	10K	4	
R4	27K	1	
R5	1K	1	
R7, R8	4k7	2	
R9	1M	1	
S1,2,3	Tactile switch NO - momentary	3	
n/a	Nano clone	1	ready built Nano version only
B1	CR1220 clip	1	
n/a	8 pin IC socket	1	only included with kit
n/a	24 pin IC socket	1	only included with kit
n/a	15 way single inline connector	2	for nano
n/a	8 way single inline connector	2	for display
n/a	20 +20 double row socket	1	for PiZero [mount on TF]
n/a	20+20 PCB double row header	1	for PiZero [mount on Pi] not included with ready built Nano version
n/a	M2.5 x 6 pan head screws	8	4 off with ready built Nano version
n/a	11mm hex brass standoff	6	4 off with ready built Nano version
PCB	TF WordClock PCB Rev 8	1	
n/a	Example WordClock templates	1pk	English (Dutch, French and Latin are an additional purchase option or free download)

Table 4, part list

11.3 Display screen shots

Figure 6 Error display

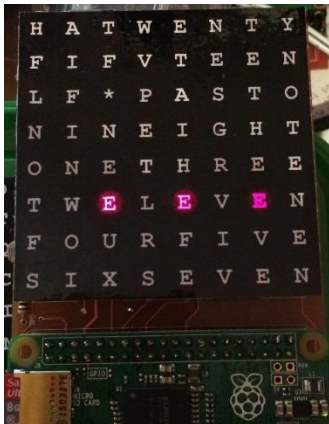


Figure 5 Version number display



11.4 Connecting other Pi's to your TF WordClock

You can use the Tempus Fugit WordClock with any of the Pi's range [except the compute module]. For the larger Pi's mounting holes are included so they can be read mounted to the PCB. The exception to this are the very early A & B Pi's these will work with the TF WordClock but do not have any mounting holes.

Using the GPIO pin assignment table in section 11.1 fit jumpers between the Pi and Tempus Fugit WordClock board.

Additionally include jumpers on

Pin 2 +5V

Pins 6 Gnd

11.5 TF WordClock Templates

The kits and ready built boards include example word templates for English, French, Dutch and Latin versions of the WordClock. These are intended as examples to 'get you going'.

- To fix the templates to the display I have found 'Pritt Stick' type glues to be really effective. Interestingly I found display mount spay glues gave poor results
- A particular advantage of using a paper glue like this is that you can get the template off and start again without damaging the display
- With version 2 s/w you can now operate the TF clock in different orientations, in which case the word template should be rotated accordingly
- The TF s/w includes an alignment mode to help position the template correctly. To put the TF into this mode hold S1 down during the start up 'bar' sequence – see section 7.3.2 for more on this.



11.5.1 Developing your own Word Templates

- The github for the TF WordClock includes word and MS publisher template files, these are a good starting point.
- 'Thin' copier paper (75gms or less) will give better results than thick 100 or 120gms paper
- For best result I recommend laminating the templates with a thermal laminator – this is how the templates included with the clock kits were made.
- An alternative is to print on to OHP paper. I have found that Inkjet OHP transparencies give better blacks than laser OHP transparencies. The Transparencies let a lot more light through given better results if you are using the TF WordClock in bright conditions.
- With care I have found you can laminate transparencies in much the same way you do paper, but it is worth noting that doing this could invalidate any warranty on your laminator
- I have found printing from MS Word and Publisher the template sizes were maintained well through the printing process.
- Printing from Adobe PDF files did not give good results

12 Index

1	Introduction – How to use this document	2
1.1	Do's and Don'ts	2
1.2	Revision History	3
2	Contents	4
3	Quick start	5
3.1	Connecting Pi / Nano to your TF WordClock	5
3.2	Fitting the RTC battery	5
3.3	Powering your TF WordClock	6
4	Build Instructions	7
4.1	Ready Built	7
4.2	Kit Built	7
4.3	Bare PCB	7
4.4	Individual build Steps	7
5	Raspberry Pi Setup	12
5.1	Assumptions	12
5.2	Configuring the RTC [I2C interface]	13
5.3	Max7219 [SPI interface]	15
5.4	Installing the TF WordClock Python s/w	16
5.5	Non-standard / Alternative setups	18
6	Arduino Nano Setup	19
7	Operating instructions	20
7.1	First Operation	20
7.2	Initialisation	20
7.3	Starting the TF WordClock applications	21
7.3.1	Forced RTC setting – Nano only	21
7.3.2	Jump to Demo / Align modes	21
7.4	Normal display mode	22
7.4.1	Setting the time	22
7.4.2	Adjusting the display brightness	22
7.4.3	Power down – Raspberry Pi only	22
8	Software Overview	23
8.1	Nano	23
8.3	Raspberry Pi – Python	24
8.3.1	Display rotations,	24
8.3.2	Blink Options,	25
9	Using the TF WordClock in other Languages	26
9.1	Dutch	26
9.2	French	26
9.3	Latin	26
10	Fault finding	27
11	Appendix, Technical Information	30
11.1	GPIO assignments	30
11.2	Parts List	30

11.3	<i>Display screen shots</i>	31
11.4	<i>Connecting other Pi's to your TF WordClock</i>	31
11.5	<i>TF WordClock Templates</i>	31
11.5.1	<i>Developing your own Word Templates</i>	32
12	Index	33