

NIST-Synced Desk Clock

Build guide for a Raspberry Pi fullscreen clock showing your local time and UTC, kept accurate by the NIST Internet Time Service.



The finished clock on the 7" display.

What it does	Shows local time in green and UTC in red, each with the date. Works in any time zone in the world: the clock uses the Pi's own time zone setting and handles daylight saving automatically. A status line along the bottom shows which NIST time server the clock is locked to and how close it is.
Raspberry Pi models	Written for the Raspberry Pi 4 used in this build. The Pi 5, Pi 3, Pi 400/500 and Zero 2 W also work; Appendix D lists what changes for each.
Skill level	Beginner. No Raspberry Pi or Linux experience needed: every command is given, ready to type.
Time	About 1 to 1½ hours, much of it waiting for downloads and updates.
Physical assembly	Not covered here. Follow the instructions that come with the SmartPi Touch Pro case and display.

How it works

The Raspberry Pi connects to your Wi-Fi. A built-in time service called **chrony** checks the NIST time servers (**time.nist.gov**) over the internet and keeps the Pi's clock locked to them, usually within a few thousandths of a second. The clock program reads the Pi's clock and draws it full-screen, once per second.

The Pi has no battery-backed clock, so after power-up it needs a minute or two to reach NIST. Until then the status line is red. It turns green once the time is confirmed.

Materials

Purchased from Adafruit (adafruit.com), as used in the original build:

Qty	Item	Adafruit product ID
1	Raspberry Pi 4 Model B, 4 GB RAM (2 GB is also plenty for this project)	4296
1	Official Raspberry Pi Foundation 7" Touchscreen Display (800 × 480)	2718
1	SmartPi Touch Pro case for Raspberry Pi and Official 7" Display	4951

Using a different Raspberry Pi or display? Read **Appendix D** before buying parts. A few parts and steps change, for example the power supply, the display cable or the screen orientation.

Display: use the **original** official 7" display (product 2718). The newer **Touch Display 2** does not fit the SmartPi Touch Pro case.

Also required (from Adafruit or anywhere):

Qty	Item	Notes
1	USB-C power supply, 5 V 3 A (Pi 4)	The official Raspberry Pi 4 power supply is strongly recommended. It powers both the Pi and the display through the case, and phone chargers are often too weak.
1	microSD card, 32 GB	Name brand (SanDisk, Samsung), rated A1 or A2
1	USB keyboard and USB mouse	Only needed during setup
1	USB flash drive	Any size. Used to carry the clock files to the Pi.
—	A Windows or Mac computer with an SD card reader	To prepare the microSD card. A USB card reader works if the computer has no slot.
—	Your Wi-Fi network name and password	The Pi 4 has built-in Wi-Fi.

Optional:

Item	When you need it
USB Wi-Fi adapter with Linux support	Only if the clock sits far from your router and Wi-Fi is unreliable. In this case the Pi sits behind the display's metal back plate, which shortens its Wi-Fi range. See Appendix A .
Network (Ethernet) cable	The most reliable connection, if a router or network jack is nearby. Plug it in and it works with no setup.

Software files

The clock software is three small files, normally supplied together as **nist-clock.zip**:

File	What it is
clock.py	The clock program
install.sh	One-step installer that sets up everything else
README.md	Short reference notes

Complete listings of **clock.py** and **install.sh** are in Appendix E, in case you need to recreate them.

Raspberry Pi basics for beginners

Term	What it means
Terminal	A window where you type commands. Open it from the black screen icon on the Pi's top bar.
Typing commands	Type each line exactly as shown, including capitals, spaces and symbols, then press Enter . Commands in this guide appear in gray boxes. Type them one line at a time.
sudo	Put at the start of a command that changes system settings. The Pi asks for your password the first time.
Password prompts	When Terminal asks for your password, nothing appears as you type , not even dots. That's normal: type it and press Enter.
~ (tilde)	Short for your home folder. On most keyboards it's the key left of 1, with Shift held.
Tab key	Type the start of a name and press Tab; Terminal fills in the rest. This saves typing and avoids typos.
Esc	Closes the fullscreen clock and returns you to the desktop.

Part A — On your computer

□ Step 1 — Install Raspberry Pi Imager

On your Windows or Mac computer, go to raspberrypi.com/software, then download and install **Raspberry Pi Imager**. It's the official free tool for setting up a Pi's microSD card.

□ Step 2 — Write Raspberry Pi OS to the microSD card

Put the microSD card in your computer and open Raspberry Pi Imager. Make these choices:

Imager asks for	Choose
Device	Raspberry Pi 4 (or your model, see Appendix D)
Operating system	Raspberry Pi OS (64-bit) , the recommended one with desktop . Do not pick a "Lite" version. (The Pi 3, 4 and 5 all use 64-bit.)
Storage	Your microSD card. Double-check it's the card and not another drive: everything on it will be erased.

Imager then lets you customise the settings. Fill in all of these, because it saves several steps later:

Setting	Enter
Hostname	nistclock (any short name works)
Username and password	Choose them, and write them down . You'll type the password several times.
Wi-Fi	Your network name (SSID) and password. Wi-Fi country: yours (US in the United States).
Locale	Time zone: your own. The clock displays this zone. Pick the city that matches your zone, e.g. New_York, Chicago, Denver, Los_Angeles, London. Keyboard layout: yours (us in the United States).
SSH (remote access)	Optional. Turn it on if you'd like to manage the Pi from your computer later.

Start writing. It takes 5–15 minutes. When Imager says it's finished, remove the card.

The Wi-Fi name and password are **case-sensitive**. One wrong letter means the Pi won't connect.

□ Step 3 — Copy the clock files to a USB flash drive

1. Plug the USB flash drive into your computer.
2. Unzip **nist-clock.zip**. On Windows: right-click it → **Extract All...** → Extract. On a Mac: double-click it.
3. Copy the **nist-clock** folder onto the USB drive, then eject the drive.

✓ **Check:** Open the **nist-clock** folder on the USB drive. You should see **clock.py**, **install.sh** and **README.md** directly inside. Windows often adds an extra layer: if you see *another* **nist-clock** folder inside instead, copy that inner folder to the USB drive instead.

Part B — On the Raspberry Pi

□ Step 4 — Assemble

Assemble the display, Pi and SmartiPi case using the instructions that come with them. Points that matter for this build:

- Keep the power supply **unplugged** during assembly.
- The display ribbon cable goes into the Pi's **DISPLAY** connector, **not** the one labeled CAMERA. (Pi 5 and other models: see Appendix D.)
- Insert the microSD card into the slot on the **underside** of the Pi.
- Plug the USB keyboard and mouse into the Pi's USB ports.

□ Step 5 — First power-up

Plug in the USB-C power supply **last**. The first start takes **2-5 minutes**, and the screen may go black and restart once by itself. Don't unplug it during this.

A) If the desktop appears (wallpaper and a top bar), go to Step 6.

B) If a “Welcome to Raspberry Pi” window appears instead, work through it:

Screen	What to choose
Set Country	Your country, language and time zone . The clock displays this time zone. In the US, tick Use US keyboard .
Create User	Choose a username and password, and write them down .
Select WiFi Network	Choose your network and enter its password.
Update Software	Click Skip . You'll update in Step 7.
Setup Complete	Click Restart .

A small **lightning bolt** symbol in a corner of the screen means the power supply is too weak. Use the official supply for your Pi model, or the clock may glitch or reboot at random.

❑ Step 6 — Check the internet connection

The network icon at the **top-right** of the screen should show Wi-Fi signal bars. If it doesn't, click it, choose your network and enter the password.

✓ **Check:** Open the web browser (globe icon on the top bar) and load any website. If it loads, you're online. Close the browser.

❑ Step 7 — Update the Pi's software

Open **Terminal** and type:

```
sudo apt update && sudo apt full-upgrade -y
sudo reboot
```

Enter your password when asked. The update takes **5-20 minutes** and prints lots of text, which is normal. The second line restarts the Pi. Wait for the desktop to come back.

❑ Step 8 — Copy the clock files onto the Pi

Plug the USB flash drive into the Pi. If a window pops up about a removable drive, choose **Open in File Manager**. Otherwise open the File Manager (folder icon on the top bar) and click the USB drive in the left panel.

Drag the **nist-clock** folder from the USB drive onto **Home** in the File Manager's left panel. Home is the folder named after your username. Then check it, in Terminal:

```
ls ~/nist-clock
```

✓ **Check:** It should print **README.md clock.py install.sh**.

If it prints...	Do this
nist-clock	The files are one folder too deep. Fix it with these two lines, then run ls ~/nist-clock again: <pre>mv ~/nist-clock/nist-clock/* ~/nist-clock/ rmdir ~/nist-clock/nist-clock</pre>
<i>No such file or directory</i>	The folder isn't in Home yet. Repeat the drag.

□ Step 9 — Run the installer

In Terminal, type:

```
cd ~/nist-clock
bash install.sh
```

Enter your password when asked. It takes a few minutes and works through five numbered stages:

Stage	What it does
1/5	Installs chrony (time sync) and pygame (graphics)
2/5	Points the time sync at time.nist.gov only
3/5	Makes Wi-Fi keep retrying after power-up, and turns off Wi-Fi power saving
4/5	Installs the clock and sets it to start automatically
5/5	Turns on desktop auto-login and turns off screen blanking

✓ **Check:** The last lines say **Done**. If it stops early with an error, the internet connection is the usual cause. Check Wi-Fi and run **bash install.sh** again; it's safe to repeat.

Don't type **sudo** in front of **bash install.sh**. The installer asks for the password itself when it needs it.

□ Step 10 — Test the clock and the NIST sync

Start the clock by hand:

```
python3 ~/nist-clock/clock.py
```

It should fill the screen. Within a minute or two, the bottom line should turn green and read *Synced to time-x-x.nist.gov*. The gray label above the green time names a city in your time zone, such as **NEW YORK (EDT)** or **CHICAGO (CDT)**. If it shows the wrong zone, see “Setting the time zone”. Press **Esc** to return to the desktop.

Status line	Meaning
Green	Synced to a NIST server. All good.
Amber	Synced, but to a non-NIST source, more than 0.1 s off, or the last update is old
Red	Not synced yet. Normal for the first minute or two after power-up.

To look at the NIST servers directly:

```
chronyc sources -v
```

✓ **Check:** Lines ending in **nist.gov** appear, and one of them is marked **^***, meaning that's the server in use.

□ Step 11 — Reboot and finish

```
sudo reboot
```

After the reboot, the desktop should log in by itself and the clock should fill the screen within about 30 seconds.

✓ **Check:** The clock appears on its own and the status line turns green within a couple of minutes. Unplug the keyboard, mouse and USB drive. **The clock is finished.**

If the screen goes dark after about 10 minutes, or the clock didn't start on its own, see “Auto-login and screen blanking” in Troubleshooting.

Troubleshooting

Problem	Fix
Screen stays black at power-up	Unplug power. Check the ribbon cable is fully seated in DISPLAY (not CAMERA) with its latch closed, and check the display power wires, following the SmartiPi instructions.
Picture is upside down	Raspberry menu → Preferences → Screen Configuration → right-click the screen → Orientation → Inverted → Apply.
Lightning bolt / random reboots	The power supply is too weak. Use the official supply for your model (Appendix D).
<i>bash: install.sh: No such file</i>	The files aren't where expected. Redo Step 8, then cd ~/nist-clock .
Installer stops with an error	Usually the internet connection. Check Wi-Fi and run bash install.sh again.
Clock shows “chrony not installed”	The installer didn't finish. Run it again (Step 9).
Need the desktop back	Press Esc or Q . A reboot brings the clock back.

Wi-Fi doesn't connect by itself after power-up

The installer already sets Wi-Fi to keep retrying. If the Pi still won't connect on its own, check the signal strength first:

```
nmcli -f IN-USE,SSID,CHAN,SIGNAL dev wifi
```

Look at the line marked *, which is your network. **SIGNAL** above 60 is good; below about 45, the connection will often time out at power-up. If the signal is weak, move the clock closer to the router, use an Ethernet cable, or add a USB Wi-Fi adapter (Appendix A).

If the signal is good, find the name of your Wi-Fi connection:

```
nmcli connection show
```

The NAME on the line whose TYPE is **wifi** is often something like **netplan-wlan0-YourNetwork** or **preconfigured**. Then type this as one line, putting in that name and your Wi-Fi password, and keep the quotes:

```
sudo nmcli connection modify "NAME" connection.autoconnect yes  
connection.permissions "" 802-11-wireless-security.psk-flags 0  
802-11-wireless-security.psk "YOURWIFIPASSWORD"
```

This stores the password for the whole system rather than in your user account, so the Pi can connect before anyone logs in. Also make sure the Wi-Fi country is set (use your two-letter country code, e.g. US, CA, GB):

```
sudo raspi-config nonint do_wifi_country US  
sudo reboot
```

Status line stays red (no NIST sync)

First make sure Wi-Fi is connected. Then tell the time service to try again now:

```
sudo chronyc online  
sudo chronyc refresh
```

Wait 2–3 minutes and run **chronyc sources -v**. If no nist.gov lines ever appear, or they never get a ^*, your router or internet provider may be blocking time traffic (UDP port 123). Check the router's settings or ask the provider.

Auto-login and screen blanking

The installer turns these on and off for you. If the clock doesn't come back by itself after a power cut, or the screen goes dark, set them by hand:

```
sudo raspi-config
```

Use the arrow keys and Enter: **System Options** → **Boot / Auto Login** → **Desktop Autologin**, then **Display Options** → **Screen Blanking** → **No**. Choose **Finish** and reboot. You can also use the desktop: Raspberry menu → Preferences → **Raspberry Pi**

Configuration (called **Control Centre** on newer versions).

Setting the time zone

The clock shows the time zone the Pi is set to, normally chosen in Step 2 or the Step 5 welcome window. Every world time zone is available, each with its own daylight saving rules. To check the current setting, type:

```
timedatectl
```

The **Time zone** line shows the setting, e.g. *America/Chicago (CDT, -0500)*.

To change it on the desktop:

Raspberry menu → Preferences → **Raspberry Pi Configuration** (**Control Centre** on newer versions) → **Localisation** tab → **Set Timezone** → choose your **Area** and **Location** → OK. Then reboot.

Or in Terminal, using a zone name from the table:

```
sudo raspi-config nonint do_change_timezone America/Chicago
sudo reboot
```

Where	Zone name	Where	Zone name
US Eastern	America/New_York	UK / Ireland	Europe/London
US Central	America/Chicago	Central Europe	Europe/Berlin
US Mountain	America/Denver	Eastern Canada	America/Toronto
Arizona (no DST)	America/Phoenix	Western Canada	America/Vancouver
US Pacific	America/Los_Angeles	Japan	Asia/Tokyo
Alaska	America/Anchorage	India	Asia/Kolkata
Hawaii	Pacific/Honolulu	Eastern Australia	Australia/Sydney

Pick the city that shares your zone. It doesn't have to be your own city. The clock labels the time with that city's name; to show your own label instead, such as EASTERN or HOME, set LOCAL_NAME as described next.

Customizing the clock

All settings are near the top of **clock.py**. Press **Esc** to close the clock, then open the file in Terminal:

```
nano ~/nist-clock/clock.py
```

Use the arrow keys to move to a number and type over it. Press **Ctrl+O** then Enter to save, **Ctrl+X** to exit, then **sudo reboot**. If the clock doesn't come back, run **python3 ~/nist-clock/clock.py**; it shows which line has the typo.

Setting	Default	What it does
FG_LOCAL	(0, 230, 60)	Local time and date color (green)
FG_UTC	(255, 30, 20)	UTC time and date color (red)
DIM	(130, 130, 130)	City and "UTC" label color (gray)
BG	(0, 0, 0)	Background color (black)
LOCAL_TZ	None	None = use the Pi's time zone setting (recommended). To force a zone, give its name in quotes, e.g. "Europe/London".
LOCAL_NAME	None	None = city from the time zone, e.g. NEW YORK. Or your own label in quotes, e.g. "EASTERN".
LOCAL_24H	False	False = 12-hour with AM/PM; True = 24-hour (usual outside the US)

Colors are three numbers, **(Red, Green, Blue)**, each from 0 to 255:

Color	Value	Color	Value
Red	(255, 30, 20)	Yellow	(255, 230, 0)
Green	(0, 230, 60)	Orange	(255, 120, 0)
Blue	(40, 120, 255)	Magenta	(255, 0, 200)
Cyan	(0, 220, 255)	White	(255, 255, 255)
Amber	(255, 176, 0)	Dim red (dark rooms)	(120, 0, 0)

Good to know

- **Daylight saving time** is automatic. NIST sends UTC only, and the Pi applies your time zone's own daylight saving rules, for example switching between EDT and EST in March and November. Running the Step 7 update every few months keeps those rules current.
- **Being a good NIST neighbor:** NIST asks for no more than one time request every 4 seconds. The clock asks every 16 seconds at power-up and backs off to about every 17 minutes once locked.
- **Turning it off:** to protect the microSD card, connect a keyboard and press Esc, then use Raspberry menu → Shutdown or type **sudo poweroff**, before unplugging when you can. Occasional power cuts are usually fine.

Quick reference

Command	Purpose
<code>chronyc sources -v</code>	List the NIST servers; ^* marks the one in use
<code>chronyc tracking</code>	Current offset, stratum and sync state
<code>python3 ~/nist-clock/clock.py</code>	Start the clock by hand
<code>nano ~/nist-clock/clock.py</code>	Edit the clock settings
<code>nmcli -f IN-USE,SSID,SIGNAL dev wifi</code>	Check Wi-Fi signal strength
<code>timedatectl</code>	Show the time zone and sync state
<code>sudo apt update && sudo apt full-upgrade -y</code>	Update the system
<code>sudo raspi-config</code>	Auto-login, screen blanking and other settings
<code>sudo reboot</code>	Restart
<code>sudo poweroff</code>	Shut down safely

Appendix A — Using a USB Wi-Fi adapter

Only needed if the built-in Wi-Fi is weak where the clock sits. An external adapter can sit outside the case, away from the display's metal back plate, often with a better antenna.

1. Choose an adapter that works with Linux out of the box

Many cheap adapters need drivers that are hard to install on a Pi. Look for one that's sold as **Linux or Raspberry Pi compatible**, or that uses a **MediaTek** chipset (for example MT7601U, MT7610U, MT7612U or MT7921AU); these drivers are already built in. Avoid listings that say “driver CD required” or “Windows/Mac only.” An adapter with an external or screw-on antenna, plus a short USB extension cable, gives the best range.

2. Plug it in and confirm the Pi sees it

```
nmcli device
```

A new Wi-Fi device should appear, usually **wlan1**. The built-in one is **wlan0**. If no new device appears after a minute, the adapter probably needs a driver that isn't built in. Try a different adapter, or search the web for its model name plus “Raspberry Pi”.

3. Compare the signal strength of each

```
nmcli -f SSID,CHAN,SIGNAL dev wifi list ifname wlan0
nmcli -f SSID,CHAN,SIGNAL dev wifi list ifname wlan1
```

Find your network in each list. If the adapter (**wlan1**) shows a clearly higher SIGNAL number, continue. If not, try moving the adapter higher or farther from the case using the extension cable.

4. Switch off the built-in Wi-Fi so the Pi uses the adapter

```
echo "dtoverlay=disable-wifi" | sudo tee -a /boot/firmware/config.txt
sudo reboot
```

After the reboot, the adapter takes over the name **wlan0** and normally reuses your existing Wi-Fi settings automatically. If it doesn't connect by itself, click the network icon, choose your network and enter the password, then reboot once more to confirm it connects on its own. Then run **bash ~/nist-clock/install.sh** once more so the Wi-Fi retry settings (installer stage 3/5) apply to the new connection.

5. Check

```
nmcli -f IN-USE,SSID,SIGNAL dev wifi
```

✓ **Check:** The line marked * shows a stronger signal than before.

To undo this and go back to the built-in Wi-Fi:

```
sudo sed -i '/dtoverlay=disable-wifi/d' /boot/firmware/config.txt
sudo reboot
```

Appendix B — What the installer changes

Item	Change
Packages	Installs chrony , which replaces the default time service, and python3-pygame
/etc/chrony/sources.d/nist.sources	New file: use the time.nist.gov pool (up to 4 servers), polling every 16 s to about 17 min
/etc/chrony/chrony.conf	Default pool/server lines commented out so only NIST is used; adds makestep 1 3 to correct a large error at power-up quickly
Wi-Fi connections	Auto-connect on, unlimited retries, Wi-Fi power saving off
~/nist-clock/clock.py	The clock program
~/.config/autostart/nist-clock.desktop	Starts the clock when the desktop starts
raspi-config settings	Boot to desktop with auto-login; screen blanking off

Appendix D — Other Raspberry Pi models and displays

The clock software is the same on every model: the installer, clock program and time zone setup don't change. Only the parts, and a few steps involving power, cables and the display, differ. Follow the main guide, with the changes listed for your model below.

Compatibility at a glance

Model	Works?	Display options	Power supply
Raspberry Pi 4 Model B	Yes (this build)	7" ribbon display or HDMI	5 V 3 A USB-C
Raspberry Pi 5	Yes	7" ribbon display (Pi 5 cable) or HDMI	27 W (5 V 5 A) USB-C
Raspberry Pi 3 Model B+ / B	Yes	7" ribbon display or HDMI	5 V 2.5 A micro-USB
Raspberry Pi 400 / 500	Yes	HDMI only	400: 5 V 3 A USB-C; 500: 27 W USB-C
Raspberry Pi Zero 2 W	Yes	HDMI (mini-HDMI cable)	5 V 2.5 A micro-USB
Pi 2, Pi 1, original Zero / Zero W	Not recommended	—	Too slow for the current desktop; Pi 1 and 2 have no Wi-Fi

D.1 Raspberry Pi 5

	Changes
Parts	Official 27 W USB-C power supply (5 V 5 A); a 3 A supply runs it but limits the USB ports. For the 7" display: a Raspberry Pi 5 display cable (22-pin to 15-pin, 200 mm). The cable that comes with the display won't fit. Optional: the official Pi 5 RTC battery, so the Pi keeps time while unplugged.
Step 2	Device: Raspberry Pi 5 .
Step 4	Plug the Pi 5 display cable into either connector marked CAM/DISP (the Pi 5 has no connector labeled DISPLAY). The SmartPi Touch Pro fits the Pi 5; use its Pi 5 mounting instructions.
Step 5	Use the 27 W supply. The Pi 5 has a power button, but it starts by itself when power is connected.

D.2 Raspberry Pi 3 Model B+ / 3 Model B

	Changes
Parts	5 V 2.5 A micro-USB power supply (official Pi 3 supply recommended). Everything else is the same as the Pi 4.
Step 2	Device: Raspberry Pi 3 .
Step 4	Same DISPLAY connector and ribbon cable as the Pi 4. Fits the SmartPi Touch Pro.
Steps 7 and 9	Expect the update and install to take about twice as long.

D.3 Raspberry Pi 400 / Raspberry Pi 500 (keyboard computers)

	Changes
Parts	An HDMI monitor or TV and a micro-HDMI to HDMI cable ; there's no ribbon-cable display connector and no SmartPi case. The keyboard is built in, so you only need a mouse. Power: 5 V 3 A USB-C (400) or 27 W USB-C (500).
Step 2	Device: Raspberry Pi 400 or Raspberry Pi 500 .
Step 4	Connect the monitor to the HDMI port nearest the power jack (HDMI0). Skip the display and case assembly.
Step 11	Keep the keyboard connected; it's part of the computer.

D.4 Raspberry Pi Zero 2 W

	Changes
Parts	An HDMI display with a mini-HDMI to HDMI cable or adapter. A micro-USB OTG adapter or small USB hub for the keyboard and mouse. A 5 V 2.5 A micro-USB power supply. A case for the Zero (the SmartPi doesn't fit).
Step 2	Device: Raspberry Pi Zero 2 W . Choose the OS version Imager marks Recommended for it.
Step 4	The Zero has two micro-USB ports: USB (for the keyboard and mouse hub) and PWR (power only). Don't mix them up.
Steps 5-9	Everything works but is slower; allow extra time for first boot, the update and the install.
Appendix A	A USB Wi-Fi adapter would need the OTG adapter too. Usually unnecessary: the Zero's antenna isn't behind a display.

D.5 Other displays

Display	Works?	What changes
Official Touch Display 2 (7", 720×1280)	Yes	Native orientation is portrait. After Step 6, turn it to landscape: Raspberry menu → Preferences → Screen Configuration → right-click the screen → Orientation → choose the setting that turns it landscape → Apply. Doesn't fit the SmartPi Touch Pro; use its own stand or case.
Any HDMI monitor or TV	Yes	No setup. The clock scales its text to fit any screen. Use micro-HDMI (Pi 4, 5, 400, 500; port nearest the power jack), standard HDMI (Pi 3) or mini-HDMI (Zero 2 W). Skip the SmartPi case.
Small HDMI screens (5"-10" kits)	Usually	Most work when plugged in. Some need a few lines added to the Pi's settings; follow the maker's instructions.
Small screens that plug onto the Pi's pins (SPI/GPIO, e.g. 3.5")	Not recommended	They need special drivers that often don't work with the current desktop.

D.6 Other supporting hardware

Item	Options
Case	Any case that fits your Pi and display. The SmartiPi Touch Pro fits the Pi 2, 3, 4 and 5 with the original 7" display.
microSD card	16 GB or larger; 32 GB recommended. Name brand, A1 or A2 rated.
Internet connection	Built-in Wi-Fi (all recommended models); an Ethernet cable (Pi 3 B/B+, 4, 5, 400, 500); or a USB Wi-Fi adapter (Appendix A). Any one is enough.
Keyboard and mouse	Any USB keyboard and mouse; only needed during setup. Wireless sets with a USB receiver work too.
Power supply	Always the official supply for your model (table above). Weak supplies cause glitches and random reboots.

Appendix E — Program listings

Provided in case you need to recreate the files. Type them exactly, without the line numbers on the left, and save them with the file names shown, in a folder named **nist-clock**.

install.sh

```
1  #!/bin/bash
2  # Installs the NIST clock on Raspberry Pi OS (64-bit, with desktop).
3  # The clock shows the Pi's own time zone setting plus UTC.
4  # Run as your normal user (not with sudo):  bash install.sh
5  # Safe to run again if anything goes wrong part-way.
6  set -e
7
8  APP_DIR="$HOME/nist-clock"
9  SRC_DIR="$(cd "$(dirname "$0")" && pwd)"
10
11 if [ ! -f "$SRC_DIR/clock.py" ]; then
12     echo "ERROR: clock.py must be in the same folder as install.sh ($SRC_DIR)"
13     exit 1
14 fi
15
16 echo "== 1/5  Installing chrony (time sync) and pygame (graphics) =="
17 sudo apt-get update
18 sudo apt-get install -y chrony python3-pygame
19
20 echo "== 2/5  Pointing chrony at NIST only =="
21 sudo mkdir -p /etc/chrony/sources.d
22 sudo tee /etc/chrony/sources.d/nist.sources >/dev/null <<'EOF'
23 # NIST Internet Time Service. time.nist.gov rotates through NIST's servers.
24 # NIST asks clients not to query more than once every 4 seconds, so no 'iburst'.
25 # Polls start at 16 s (fast first sync) and back off to ~17 min once locked.
26 pool time.nist.gov maxsources 4 minpoll 4 maxpoll 10
27 EOF
28 # comment out the stock Debian/Pi pool & server lines so only NIST is used
29 sudo sed -i -E 's/^(pool|server)[[:space:]]/##&/' /etc/chrony/chrony.conf
30 grep -q '^sourcedir /etc/chrony/sources.d' /etc/chrony/chrony.conf || \
31     echo 'sourcedir /etc/chrony/sources.d' | sudo tee -a /etc/chrony/chrony.conf >/dev/null
32 grep -q '^makestep' /etc/chrony/chrony.conf || \
33     echo 'makestep 1 3' | sudo tee -a /etc/chrony/chrony.conf >/dev/null
34 sudo systemctl enable chrony
35 sudo systemctl restart chrony
36
37 echo "== 3/5  Making Wi-Fi reconnect reliably after power-up =="
38 # keep retrying forever instead of giving up, and turn off Wi-Fi power saving
39 if command -v nmcli >/dev/null; then
40     nmcli -t -f NAME,TYPE connection show | while IFS=: read -r name type; do
41         if [ "$type" = "802-11-wireless" ]; then
42             sudo nmcli connection modify "$name" connection.autoconnect yes \
43                 connection.autoconnect-retries 0 802-11-wireless.powersave 2 \
44                 && echo "    tuned Wi-Fi connection: $name" || true
45         fi
46     done
47 fi
48
49 echo "== 4/5  Installing clock to $APP_DIR and starting it at login =="
50 mkdir -p "$APP_DIR"
51 if [ "$SRC_DIR" != "$(cd "$APP_DIR" && pwd)" ]; then
52     cp "$SRC_DIR/clock.py" "$APP_DIR/clock.py"
53 fi
54 chmod +x "$APP_DIR/clock.py"
55 mkdir -p "$HOME/.config/autostart"
56 cat > "$HOME/.config/autostart/nist-clock.desktop" <<EOF
57 [Desktop Entry]
58 Type=Application
59 Name=NIST Clock
60 Exec=/usr/bin/python3 $APP_DIR/clock.py
61 X-GNOME-Autostart-enabled=true
62 EOF
63
64 echo "== 5/5  Desktop auto-login on, screen blanking off =="
65 sudo raspi-config nonint do_boot_behaviour B4 || echo "    (set auto-login by hand)"
66 sudo raspi-config nonint do_blanking 1 || echo "    (turn off blanking by hand)"
67
68 echo
```

```

69 echo "Done."
70 echo "   Time zone shown:      ${timedatectl show -p Timezone --value 2>/dev/null}   (to change: see guide)"
71 echo "   Test the clock now:   python3 $APP_DIR/clock.py   (Esc or Q to quit)"
72 echo "   Check NIST sync:      chronyc sources -v"
73 echo "   Then reboot:          sudo reboot"

```

clock.py

```

1  #!/usr/bin/env python3
2  """
3  NIST-synced fullscreen clock for Raspberry Pi (built on a Pi 4; other models work too)
4  Shows local time (the Pi's own time zone setting) and UTC, plus a live
5  time-sync status line from chrony.
6
7  The system clock is disciplined to time.nist.gov by chrony (see nist.sources);
8  this program just displays it, so the time shown is only as good as the sync,
9  and the status bar tells you how good that is.
10
11 Exit with Esc or Q.
12 """
13 import subprocess
14 import threading
15 import time
16 import os
17 from datetime import datetime, timezone
18 from zoneinfo import ZoneInfo
19
20 import pygame
21
22 # ----- settings -----
23 # Local time zone: None = use the Pi's own time zone setting (recommended).
24 # To force one, give its name, e.g. "America/Chicago" or "Europe/London".
25 LOCAL_TZ = None
26 # Label above the local time: None = city from the time zone, e.g. "NEW YORK".
27 LOCAL_NAME = None
28 LOCAL_24H = False          # False = 12-hour with AM/PM, True = 24-hour
29 STATUS_INTERVAL = 10      # seconds between chrony status checks
30
31 BG          = (0, 0, 0)
32 FG_LOCAL    = (0, 230, 60) # green
33 FG_UTC      = (255, 30, 20) # red
34 DIM         = (130, 130, 130)
35 OK          = (0, 200, 80)
36 WARN       = (230, 180, 0)
37 BAD        = (230, 40, 40)
38 FONT       = "dejavusansmono"
39 # -----
40
41
42
43 def local_zone():
44     """Return (tzinfo, zone name) for the local time zone."""
45     if LOCAL_TZ:
46         return ZoneInfo(LOCAL_TZ), LOCAL_TZ
47     try: # /etc/localtime -> /usr/share/zoneinfo/America/New_York
48         path = os.path.realpath("/etc/localtime")
49         name = path.split("zoneinfo/", 1)[1]
50         return ZoneInfo(name), name
51     except Exception:
52         return datetime.now().astimezone().tzinfo, ""
53
54
55 def zone_label(name):
56     """Turn "America/New_York" into "NEW YORK"; fall back to "LOCAL"."""
57     if LOCAL_NAME:
58         return LOCAL_NAME
59     city = name.rsplit("/", 1)[-1].replace("_", " ").upper()
60     if not city or city in ("UTC", "UCT", "UNIVERSAL", "ZULU") or city.startswith("GMT"):
61         return "LOCAL"
62     return city
63
64
65 _status = {"text": "Checking time sync...", "color": DIM}
66 _lock = threading.Lock()
67

```

```

68
69 def _fmt_offset(sec):
70     a = abs(sec)
71     sign = "+" if sec >= 0 else "-"
72     if a < 1e-3:
73         return f"{sign}{a * 1e6:.0f} \u00b5s"
74     if a < 1:
75         return f"{sign}{a * 1e3:.2f} ms"
76     return f"{sign}{a:.2f} s"
77
78
79 def read_chrony():
80     """Return (text, color) describing the current sync state."""
81     try:
82         out = subprocess.run(["chronyc", "-c", "tracking"], capture_output=True,
83                               text=True, timeout=5).stdout.strip()
84         f = out.split(",")
85         name, stratum = f[1], int(f[2])
86         ref_time, offset, leap = float(f[3]), float(f[4]), f[-1]
87     except FileNotFoundError:
88         return "chrony not installed - time NOT verified", BAD
89     except Exception:
90         return "Unable to read sync status from chrony", BAD
91
92     if leap.lower().startswith("not") or stratum == 0 or ref_time == 0:
93         return "NOT SYNCHRONIZED - waiting for NIST", BAD
94
95     age = time.time() - ref_time
96     text = (f"Synced to {name} \u2022 stratum {stratum} \u2022 "
97            f"offset {_fmt_offset(offset)} \u2022 last update {age:.0f} s ago")
98     color = OK
99     if "nist" not in name.lower():
100         color = WARN
101     if abs(offset) > 0.1 or age > 3000:
102         color = WARN
103     return text, color
104
105
106 def status_worker():
107     while True:
108         text, color = read_chrony()
109         with _lock:
110             _status["text"], _status["color"] = text, color
111         time.sleep(STATUS_INTERVAL)
112
113
114 def fit_font(sample, max_w, start_size, bold=True):
115     size = start_size
116     while size > 10:
117         font = pygame.font.SysFont(FONT, size, bold=bold)
118         if font.size(sample)[0] <= max_w:
119             return font
120         size -= 2
121     return pygame.font.SysFont(FONT, size, bold=bold)
122
123
124 def main():
125     pygame.init()
126     screen = pygame.display.set_mode((0, 0), pygame.FULLSCREEN)
127     pygame.display.set_caption("NIST Clock")
128     pygame.mouse.set_visible(False)
129     W, H = screen.get_size()
130     tz, tz_name = local_zone()
131     place = zone_label(tz_name)
132
133     big = fit_font("88:88:88", W * 0.80, int(H * 0.26))
134     label = fit_font(max(f"{place} (WWWW)", "EASTERN (EDT) 88", key=len),
135                      W * 0.90, int(H * 0.055))
136     datef = fit_font("Wednesday, September 30, 2026", W * 0.90, int(H * 0.05), bold=False)
137     status_sample = ("Synced to time-x-x.nist.gov \u2022 stratum 1 \u2022 "
138                     "offset +888.88 ms \u2022 last update 8888 s ago")
139     small = fit_font(status_sample, W * 0.96, int(H * 0.03), bold=False)
140     suffix = pygame.font.SysFont(FONT, int(big.get_height() * 0.30), bold=True)
141
142     threading.Thread(target=status_worker, daemon=True).start()
143

```

```

144 def draw_block(top, title, time_str, ampm, date_str, color):
145     y = top
146     surf = label.render(title, True, DIM)
147     screen.blit(surf, surf.get_rect(midtop=(W // 2, y)))
148     y += surf.get_height()
149
150     t = big.render(time_str, True, color)
151     s = suffix.render(ampm, True, color) if ampm else None
152     total_w = t.get_width() + (s.get_width() + W * 0.01 if s else 0)
153     x = (W - total_w) / 2
154     screen.blit(t, (x, y))
155     if s:
156         # sit the AM/PM roughly on the digit baseline
157         screen.blit(s, (x + t.get_width() + W * 0.01,
158             y + big.get_ascent() - suffix.get_ascent()))
159     y += t.get_height() * 0.92
160
161     d = datef.render(date_str, True, color)
162     screen.blit(d, d.get_rect(midtop=(W // 2, y)))
163
164     running, last = True, None
165     while running:
166         for e in pygame.event.get():
167             if e.type == pygame.QUIT:
168                 running = False
169             elif e.type == pygame.KEYDOWN and e.key in (pygame.K_ESCAPE, pygame.K_q):
170                 running = False
171
172         now = time.time()
173         sec = int(now)
174         if sec != last:
175             last = sec
176             utc = datetime.fromtimestamp(sec, timezone.utc)
177             loc = utc.astimezone(tz)
178
179             if LOCAL_24H:
180                 l_time, l_ampm = loc.strftime("%H:%M:%S"), ""
181             else:
182                 l_time, l_ampm = loc.strftime("%I:%M:%S"), loc.strftime("%p")
183
184             screen.fill(BG)
185             draw_block(int(H * 0.04), f"{place} ({loc.tzname()})",
186                 l_time, l_ampm, loc.strftime("%A, %B %d, %Y"), FG_LOCAL)
187             draw_block(int(H * 0.48), "UTC",
188                 utc.strftime("%H:%M:%S"), "", utc.strftime("%A, %B %d, %Y"), FG_UTC)
189
190             with _lock:
191                 st_text, st_color = _status["text"], _status["color"]
192                 st = small.render(st_text, True, st_color)
193                 screen.blit(st, st.get_rect(midbottom=(W // 2, H - int(H * 0.015))))
194             pygame.display.flip()
195
196         # wake up just after the next second boundary (~1 ms rollover latency)
197         time.sleep(max(0.001, min(0.05, 1.0 - (time.time() % 1.0))))
198
199     pygame.quit()
200
201
202 if __name__ == "__main__":
203     main()

```