Thanks for purchasing a PLC14500 - Nano. Whether you got the full kit, or just the board and will be sourcing the components yourself, what awaits you is a retro-style trainer board based on the Motorola MC14500 1-bit ICU. The board has abundant LEDs that show the status of the system buses and registers. This, combined with the possibility to run the software step by step, will give you a great deal of insight into how your programs are executing. The slow clock mode will make for a mesmerising light show. It won't fail to impress your guests when the board is parked on a desk, waiting for your next coding adventure.

In this guide you will find all the information you need to assemble and test the board as well as to move your first steps to program it.

This project is Open Source Hardware Certified because I believe people can learn a lot not only by using the finished product, but by being able to see how it was designed and how the software that supports it works.

If you feel you have something to contribute, need clarifications or would like to get in touch for any other reason, feel free to do so on the “Discussions” section of the project’s GitHub page (github.com/nicolacimmino/PLC-14500).

OSHW
PL000011

_____

Nicola Cimmino

Assembling the board

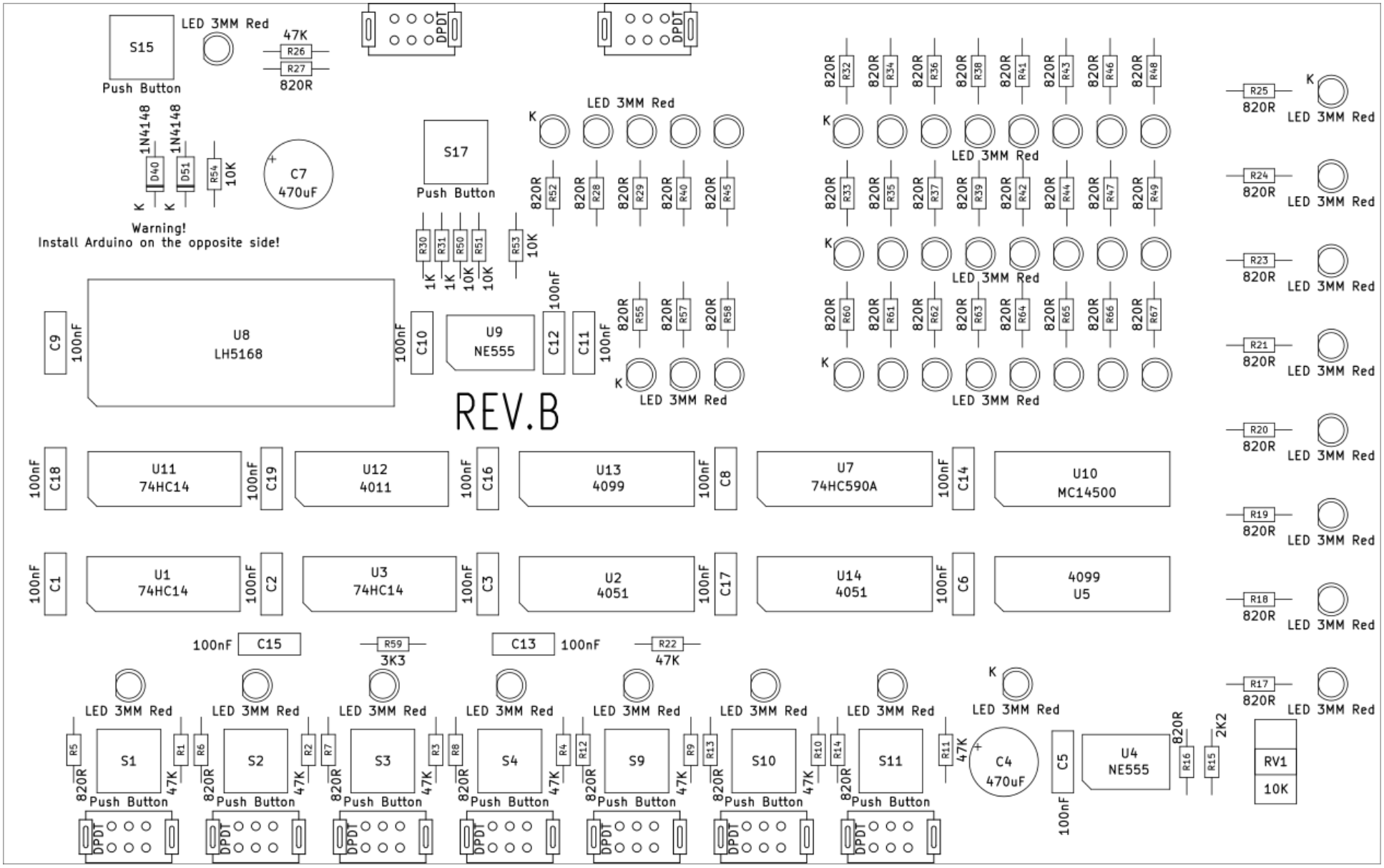
To keep the assembled board visually free from clutter, and to give more prominence to the labels relevant in regular use, it was chosen to hide the component reference numbers on the silkscreen. Where possible these are on the silkscreen under the component itself but, where not possible, these have been omitted completely. For this reason you won't be able to see all the component values on the silkscreen. Please refer to the assembly plan included in the next page.

NOTE: Double check that the the revision label on the assembly plan matches the one at the **back** of the board (eg. "REV.B"). If you received for any reason an assembly plan for the wrong board revision, you can find the right one on the GitHub repo under "/board").

Before starting the assembly double check you received all components as per bills-of-materials below.

You will need a soldering iron with a medium/fine tip and solder (anything not too thin will do). All components are through-hole so you won't need very specialised tools. You could use a multimeter to help you recognize the resistors if you are not familiar with the colour bands code.

It will be useful to have some IPA (Isopropyl Alcohol) to clean the board soldering work at the end.


S15
Push Button
LED 3MM Red
47K
R26
R27
820R
DPDT
DPDT
1N4148
D40
1N4148
D51
R54
10K
K
K
C7
470uF
Warning!
Install Arduino on the opposite side!
S17
Push Button
LED 3MM Red
K
820R R52
820R R28
820R R29
820R R40
820R R45
R30 1K
R31 1K
R50 10K
R51 10K
R53 10K
820R R32
820R R34
820R R36
820R R38
820R R41
820R R43
820R R46
820R R48
K
LED 3MM Red
820R R33
820R R35
820R R37
820R R39
820R R42
820R R44
820R R47
820R R49
K
LED 3MM Red
820R R60
820R R61
820R R62
820R R63
820R R64
820R R65
820R R66
820R R67
K
LED 3MM Red
R25 820R
R24 820R
R23 820R
R21 820R
R20 820R
R19 820R
R18 820R
R17 820R
K
LED 3MM Red
C9 100nF
U8
LH5168
100nF C10
U9
NE555
C12 100nF
C11 100nF
820R R55
820R R57
820R R58
K
LED 3MM Red
REV.B
100nF C18
U11
74HC14
100nF C19
U12
4011
100nF C16
U13
4099
100nF C8
U7
74HC590A
100nF C14
U10
MC14500
100nF C1
U1
74HC14
100nF C2
U3
74HC14
100nF C3
U2
4051
100nF C17
U14
4051
100nF C6
4099
U5
100nF C15
R59
3K3
C13 100nF
R22
47K
LED 3MM Red
LED 3MM Red
LED 3MM Red
LED 3MM Red
LED 3MM Red
LED 3MM Red
LED 3MM Red
K
LED 3MM Red
R5 820R
S1
R1 47K
Push Button
R6 820R
S2
R2 47K
Push Button
R7 820R
S3
R3 47K
Push Button
R8 820R
S4
R4 47K
Push Button
R12 820R
S9
R9 47K
Push Button
R13 820R
S10
R10 47K
Push Button
R14 820R
S11
R11 47K
Push Button
DPDT
DPDT
DPDT
DPDT
DPDT
DPDT
DPDT
C4
470uF
C5
100nF
U4
NE555
820R R16
2K2 R15
RV1
10K

Ref	Qnty	Value
C1, C2, C3, C5, C6, C8, C9, C10, C11, C12, C13, C14, C15, C16, C17, C18, C19	17	100nF
C4, C7	2	470uF
D1, D2, D3, D4, D5, D6, D7, D8, D9, D10, D11, D12, D13, D14, D15, D16, D17, D18, D19, D20, D21, D22, D23, D24, D25, D26, D27, D28, D29, D30, D31, D32, D33, D34, D35, D36, D37, D38, D39, D41, D42, D43, D44, D45, D46, D47, D48, D49, D50	49	LED 3MM Red
D40, D51	2	1N4148
R1, R2, R3, R4, R9, R10, R11, R22, R26	9	47K
R5, R6, R7, R8, R12, R13, R14, R16, R17, R18, R19, R20, R21, R23, R24, R25, R27, R28, R29, R32, R33, R34, R35, R36, R37, R38, R39, R40, R41, R42, R43, R44, R45, R46, R47, R48, R49, R52, R55, R57, R58, R60, R61, R62, R63, R64, R65, R66, R67	49	820R
R15	1	2K2
R30, R31	2	1K
R50, R51, R53, R54	4	10K
R59	1	3K3
RV1	1	10K
S1, S2, S3, S4, S9, S10, S11, S15, S17	9	Push Button
S5, S6, S7, S8, S12, S13, S14, S16, S18	9	Switch DPDT
U1, U3, U11	3	74HC14
U2, U14	2	4051
U4, U9	2	NE555
U5, U13	2	4099
U6	1	Arduino Nano
U7	1	74HC590A
U8	1	LH5168
U10	1	MC14500
U12	1	4011

Start soldering from the lowest profile components such as resistors and diodes. Pay attention to the orientation of the 1N4148 diodes, the cathode is marked by a band on the body of the component, which needs to match the band on the silkscreen. Resistors are unpolarized, so they can be inserted either way.

Move on then to the push-buttons. Due to slight tolerances and the springiness of the pins you might need to slightly spread the pins for them to fit snugly in the holes. After the buttons it’s a good idea to do the slider switches. To ensure they are flush with the board, solder only one pin of each first. Then rework them one by one by pressing with a finger on the top of the body of the switch and re-heating the solder point until the switch sits flat on the board. You can then proceed to solder all the other pins. **NOTE**: to prevent thermal damage to the switches it’s a good idea to solder one pin at the time of each switch, then move on to the next switch and so on. This will leave time for them to cool down.

Next are the ICs. Pay close attention to the markings on the body, helping yourself with a magnifying glass where needed. Many chips have the same amount of pins but are different! Also pay attention to the orientation of the chip. The side with pin 1 has a half-circle indent that needs to be aligned with the side with pin 1 on the silk screen (the cut-off corner). If you look at the board with the silk screen oriented for normal reading, all chips’ pin 1s are on the left. As for the switches, solder only one pin of each IC, rework it while pressing on the body to ensure it sits flat, and then solder the remaining pins alternating between different chips to avoid overheating.

Proceed with the LEDs. All cathodes are towards the left side of the board, the cathode is the shortest of the two leads. To ensure the LEDs sit flush and are nicely aligned, solder a single pin of each first and then rework them one by one while

holding the board in your hand and pressing with a finger on the LED.

Next are the 100nF polyester capacitors. These are unpolarized so the direction doesn’t matter. Move on then to the 470uF electrolytic capacitors. These are polarised so you need to pay attention to the orientation. The negative terminal is marked by a grey band, with a “-” sign on the body.

Last on this side of the board is the variable resistor. This can fit in only one position so it should be easy to orient.

You can now move to the back of the board where the only component is the Arduino Nano that provides the USB interface for the board and its bootloader. Assemble the pin headers on the Arduino board first, then sit the board with the headers into the matching pin holes and solder from the components side.

Testing the Board

If you purchased the full kit the Arduino comes pre-loaded with the bootloader and flashed with the smoke-test application. If you opted for sourcing the components yourself, please refer to the “Getting Started” section on the GitHub README.md in the root folder of the repository (https://github.com/nicolacimmino/PLC-14500/README.md) for instructions on how to load the bootloader into the Arduino and compile and flash your PLC14500 programs.

The smoke test program pre-flashed is designed to provide a simple way to verify the main features of the board and validate the assembly. The source code is reported in the next page and contains, in the comments, the steps needed to perform each step of the test.

```
.board=PLC14500-Nano
; ************************************************
; Prepare:
; All inputs off except IN6 (master switch).
; ************************************************

IEN IN6   ; IN6 acts as master switch
OEN IN6   ; to enable/disable all I/O

; ************************************************
; Test: Input and outputs are working correctly

LD  IN0   ; Load IN0,
AND IN1   ; logical AND it with IN1
STO OUT0  ; and show the result in OUT0.

; Expect: OUT0 is on only when both IN0 and IN1 are on.
; ************************************************

; ************************************************
; Test: TMR0 is working correctly

LD  IN2   ; Load IN2
STO OUT7  ; use it to trigger TMR0.

; Expect: Clicking IN2 on turns on IN7 (TMR0 output)
;   and, after a few seconds, IN7 returns off.
; ************************************************

; ************************************************
; Test: Scratchpad RAM is working correctly.

LD  IN3    ; Copy IN3 to SPR0
STO SPR0

LD  SPR0   ; Copy SPR0 to SPR1
STO SPR1

; Expect: Clicking IN3 turns on SPR0 and SPR1
; ************************************************

JMP 0     ; Repeat.
```

Next Steps

Now that your board is assembled and tested, go ahead to the releases folder on GitHub to download the latest tools (https://github.com/nicolacimmino/PLC-14500/releases).

You can then follow the instructions on the “Getting Started” section on the GitHub README.md in the root folder of the repository to learn how to write, assemble, and flash your own programs.

The “Programmers Guide” is your next step as it provides more in depth information on the board programming model as well as practical examples to try and modify.