

KOCAELİ UNIVERSITY ENGINEERING FACULTY



**ARDUINO LCD KEYPAD SHIELD
REFERENCE DOCUMENT**

**Furkan YARDIMCI
210207089**

**DEPARTMENT: Electronic and Communacition Engineering
TEACHER: Associate Profesör Dr. ANIL ÇELEBİ**

KOCAELİ 2022

TABLE OF CONTENTS

TABLE OF CONTENTS	2
LIST OF FIGURES	2
LIST OF TABLES	2
PURPOSE	3
1. INTRO/OVERVIEW	3
1.1. FEATURES	3
2. LCD DISPLAY	4
2.1. PINCONFIGURATION	4
2.2. CONTROL AND DISPLAY COMMANDS	5
2.3. INITIALIZATION OF LCD	6
2.4. USAGE OF 4-BIT MODE	6
2.5. ALPHA NUMERIC TABLE	7
3. VERILOG HDL	8
3.1 XDC FILE	8
3.2. LCD.V	8-15
4. SCHEMATICS	16
5. PCBS	17
6. FINAL VIEW OF THE PROJECT	18
7. REFERENCES	19

LIST OF IMAGES

Img 1: LCD keypad shield	3
Img 2: 2x16 LCD display	4
Img 3: 4-bit mode working process	6
Img 4: Alphanumeric table	7
Img 5: Schematic1	16
Img 6: Schematic2	16
Img 7: PCB1	17
Img 7: PCB2	17
Img 8: Front view of the project	18
Img 9: Back view of the project	18

LIST OF TABLES

Table 1: Pins of LCD display CONTENTS	4
Table 2: LCD instructions	5
Table 3: LCD initializations	6
Table 4: LCD instructions used	6

PURPOSE

The purpose of this document is to provide a document containing the necessary features about the design and use of LCD Keypad shield, prepared by Kocaeli University Students FURKAN YARDIMCI.

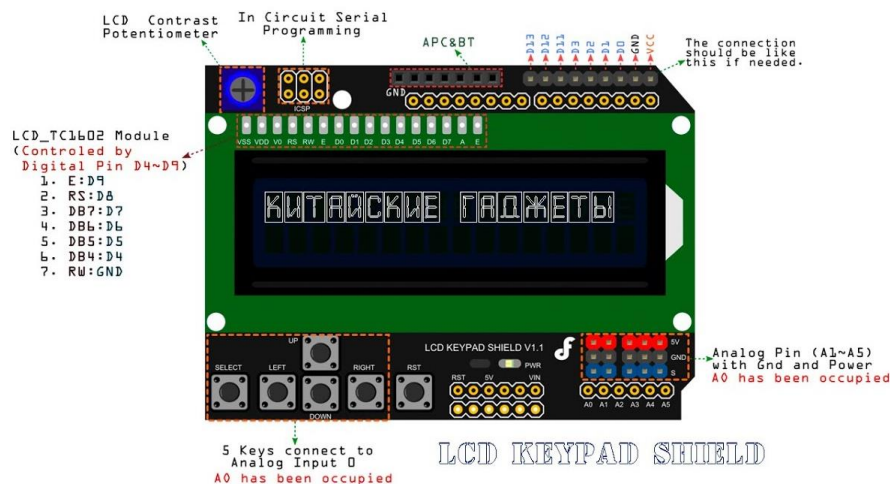
1. INTRO/OVERWIEV

LCD Keypad shield for Arduino or Freeduino board. It includes a 2x16 LCD display and 6 momentary push buttons. Pins 4, 5, 6, 7, 8, 9 and 10 are used to interface with the LCD. Analog Pin 0 is used to read the push buttons. The LCD shield supports contrast adjustment and backlight on/off functions. It also expands analog pins for easy analog sensor reading and display. The LCD Keypad shield is developed for Arduino compatible boards, to provide a user-friendly interface that allows users to go through the menu, make selections etc. It consists of a 1602 white character blue backlight LCD. The keypad consists of 5 keys — select, up, right, down and left. To save the digital IO pins, the keypad interface uses only one ADC channel. The key value is read through a 5 stage voltage divider.

1.1 FEATURES:

- Operating Voltage:5V
- 5 Push buttons to supply a custom menu control panel
- RST button for resetting Arduino program (in our case PYNQ-Z2)
- Integrate a potentiometer for adjusting the backlight
- Expanded available I/O pins
- Expanded Analog Pinout with standard DFRobot configuration for fast sensor extension

The buttons for the right, left, front and back on the module are connected to the A0 pin in common. You can understand which button was pressed by processing the data coming to the A0 pin. In this way, you will also save on pins. In addition, LCD works in 4-bit mode. Detailed information about 4-bit mode is given in the next section.



Img 1: LCD keypad shield

2. LCD DISPLAY

LCD (Liquid Crystal Display) screen is an electronic display module and find a wide range of applications. A 16x2 LCD display is very basic module and is very commonly used in various devices and circuits. These modules are preferred over seven segments and other multi segment LEDs. The reasons being: LCDs are economical; easily programmable; have no limitation of displaying special & even custom characters (unlike in seven segments), animations and so on. A 16x2 LCD means it can display 16 characters per line and there are 2 such lines. In this LCD each character is displayed in 5x7 pixel matrix. This LCD has two registers, namely, Command and Data.

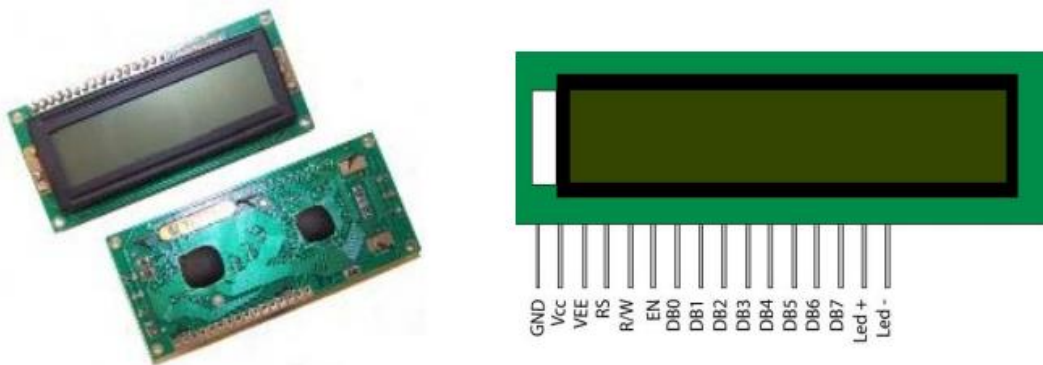


Image 2: 2x16 LCD Display

2.1 PIN CONFIGURATION

SM PMOD Reference Designator	Function
Pin 1 (Vss)	Function as Ground Terminal.
Pin 2 (Vcc)	Function as Positive Supply (2.7V to 5.5V).
Pin 3 (Vdd)	Function as Contrast adjustment (Ground to Vcc).
Pin 4 (RS)	Function as Register Select (If 0 is refer to Instruction Register and if 1 is refer to Data Register).
Pin 5 (R/W)	Its function to Read or Write Signal (if 1 mean to Read and if 0 mean to Write).
Pin 6 (E)	Function as Enable.
Pin 7 to Pin 14 (DB0 – DB7)	: Refer to Bi-directional data bus, data transfer is performed one, thru DB0 to DB7, in this case of interface data length is 8- bits; and twice, through DB4 to DB7 in this case of interface data length is 4- bits (Upper nibble first and then Lower nibble).
Pin 15 (K)	Function to Back light LED cathode terminal.
Pin 16 (A):	Function to Back light LED anode terminal.

Table 1: pins of LCD display

2.2 CONTROL AND DISPLAY COMMANDS

There is a character LCD controller of Hitachi company named HD44780 on most character LCDs available in the market. This controller acts as a bridge between the LCD and the FPGA. In other words, we do not directly interfere with the pixels on the LCD with the FPGA. We ensure that the characters we want are displayed through the controller. Since the HD44780 is a general-purpose controller, most character LCD manufacturers use this controller in their LCDs of various sizes and features.

3 bits are very important for sent data or instruction

RS: Register Select pin. In the logic 0 state, a command is sent to the HD44780 from the bus. If logic 1 is set, data is written or read from the data bus to the HD44780. It is necessary to write or receive commands to make adjustments to the HD44780, and to send or receive character data.

R/W: Read write pin. In the logic 1 state, reading is taken from HD44780. If the logic is 0, the HD44780 is written to. Since the HD44780 is generally written to, this pin is usually connected directly to gnd in applications.

EN: It is the enable pin. In case of logic 1, read-write operation is performed to HD44780. Cannot be done in the logic 0 state. D0:D7: The bus of HD44780 controller is 8 bits wide, used for reading, write operations.

To send instructions to LCD via controller, RS and R/W bits must be 0 and to send data RS must be 1. In this mode, we can send commands to LCD's instruction register. The commands are 8 bit wide and given in the list below.

Sr.No.	Hex Code	Command to LCD instruction Register
1	01	Clear display screen
2	02	Return home
3	04	Decrement cursor (shift cursor to left)
4	06	Increment cursor (shift cursor to right)
5	05	Shift display right
6	07	Shift display left
7	08	Display off, cursor off
8	0A	Display off, cursor on
9	0C	Display on, cursor off
10	0E	Display on, cursor blinking
11	0F	Display on, cursor blinking
12	10	Shift cursor position to left
13	14	Shift the cursor position to the right
14	18	Shift the entire display to the left
15	1C	Shift the entire display to the right
16	80	Force cursor to the beginning (1st line)
17	C0	Force cursor to the beginning (2nd line)
18	38	2 lines and 5×7 matrix (8bit mode)
19	28	2 lines and 5×7 matrix (4bit mode)

Table 2: LCD instructions

2.3 INITIALIZATION OF LCD

According to datasheet some process must be followed beginning of the device start to function properly. This process steps are given below. To send instructions RS bit should be 0 and to write R/W bit should be 0 as mentioned above and enable pin must be high at least 230ns to send or write any data to lcd. Waits are given due to this information.

1	EN = 0	Wait 15ms or longer before enable the device.
2	EN = 1	Send 0x3 for 240ns.
3	EN = 0	Wait 4.1ms or longer.
4	EN = 1	Send 0x3 for 240ns.
5	EN = 0	Wait 100us or longer.
6	EN = 1	Send 0x3 for 240ns.
7	EN = 0	Wait 40us or longer.
8	EN = 1	Send 0x2 for 240ns.
9	EN = 0	Wait 40us or longer.

Table 3: LCD initialization

After initialization process, we can send data or instruction to LCD. First, we will send instructions. Used instructions are listed below.

1	0E	Display on, cursor blinking
2	06	Increment cursor (shift cursor to right)
3	80	Force cursor to the beginning (1st line)
4	28	2 lines and 5×7 matrix (4bit mode)

Table 4: LCD instructions used

2.4 USAGE OF 4-BIT MODE

8 bit data length is expensive in terms of pin usage so LCD used in 4 bit mode in this project. To do this 0x28 instruction must be send to LCD when RS and R/W bits are 0. In this mode, data is sent in nibbles. The MSB nibble must be sent first, followed by the LSB nibble. Regardless of whether the information sent is data or instructions, a certain process must be followed. After sending MSB nibble, wait 1 microsecond and then send LSB nibble and wait at least 40 microseconds at the end of each 8-bit data packet. This process must be followed to send instruction or data to LCD.

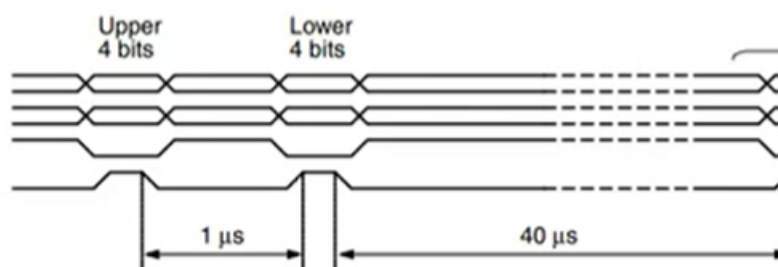


Image 3: 4-bit mode working process

2.5 ALPHA NUMERIC TABLE

The 2x16 LCD display has the power to write $2 \times 16 = 32$ characters in total. It uses a 5x7 matrix to write each character to the screen. The upper and lower nibble values of these characters are given below. To be able to read these characters on the screen, RS = 1 and R/W = 0 and when RS = 1, we send data to LCD's data register as mentioned early.

Lower 4 Bits \ Upper 4 Bits	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111	
xxxx0000	CG RAM (1)			0	1	P	`	P				-	9	3	α	p	
xxxx0001	(2)			!	1	A	Q	a	q			。	7	チ	△	ä	q
xxxx0010	(3)			"	2	B	R	b	r			「	イ	ツ	×	β	θ
xxxx0011	(4)			#	3	C	S	c	s			」	ウ	テ	ε	ε	∞
xxxx0100	(5)			\$	4	D	T	d	t			、	エ	ト	†	μ	Ω
xxxx0101	(6)			%	5	E	U	e	u			・	オ	ナ	1	σ	Ü
xxxx0110	(7)			&	6	F	V	f	v			ヲ	カ	ニ	ヨ	ρ	Σ
xxxx0111	(8)			'	7	G	W	g	w			ア	キ	ヌ	ラ	g	π
xxxx1000	(1)			(8	H	X	h	x			ィ	ク	ネ	リ	フ	×
xxxx1001	(2))	9	I	Y	i	y			ウ	ケ	ノ	ル	”	γ
xxxx1010	(3)			*	:	J	Z	j	z			エ	コ	ハ	レ	j	〒
xxxx1011	(4)			+	;	K	[k	{			オ	サ	ヒ	ロ	*	⌘
xxxx1100	(5)			,	<	L	¥	l				カ	シ	フ	ワ	⊕	⌘
xxxx1101	(6)			-	=	M]	m	}			ユ	ズ	ヘ	ン	も	÷
xxxx1110	(7)			.	>	N	^	n	÷			ヨ	セ	ホ	”	ñ	
xxxx1111	(8)			/	?	O	_	o	€			ッ	ソ	マ	”	ö	■

Img 4: Alphanumeric table

3. VERILOG HDL:

Procedural design was used for the initial and control processes of the LCD mentioned in the previous section. The always block is sensitive to the rising edge of the clock signal and is triggered on each rising edge. Appropriate waiting times are given to the datasheet of the LCD between each state, and these waiting times are given specifically for the PYNQ-Z2 card. For different cards, these times should be recalculated depending on the card's clock speed. The code is as follows.

3.1 XDC FILE

```

1  ## Arduino GPIO
2  set_property -dict {PACKAGE_PIN V15 IOSTANDARD LVCMOS33} [get_ports {data[0]}]
3  set_property -dict {PACKAGE_PIN T15 IOSTANDARD LVCMOS33} [get_ports {data[1]}]
4  set_property -dict {PACKAGE_PIN R16 IOSTANDARD LVCMOS33} [get_ports {data[2]}]
5  set_property -dict {PACKAGE_PIN U17 IOSTANDARD LVCMOS33} [get_ports {data[3]}]
6  set_property -dict {PACKAGE_PIN V17 IOSTANDARD LVCMOS33} [get_ports {RS}]
7  set_property -dict {PACKAGE_PIN V18 IOSTANDARD LVCMOS33} [get_ports {EN}]
8
9  # Clock signal 125 MHz
10 set_property -dict { PACKAGE_PIN H16 IOSTANDARD LVCMOS33 } [get_ports { CLK }];
11 create_clock -add -name sys_clk_pin -period 8.00 -waveform {0 5} [get_ports { CLK }];

```

3.2 LCD.V

```

1  `timescale 1ns / 1ps
2
3  module LCD(input CLK,
4      output [3:0] data,
5      output reg RS, EN);
6
7      reg [3:0] lcd_cmd;
8      reg [5:0] state = 0;
9      reg [25:0] count=0;
10
11      assign data = lcd_cmd;
12
13      always @(posedge CLK) begin
14          case(state)
15              //init işlemlerinin başlangıcı. Bu aşama raporda detaylı olarak anlatılmıştır.
16              0:begin
17                  EN <= 0;
18                  RS <= 0;
19                  if(count == 1875000)
20                      begin
21                          count <= 0;
22                          state <= state + 1;
23                      end
24                  else
25                      count <= count+1;
26                  end
27
28              1: begin
29                  EN <= 1;
30                  lcd_cmd <= 4'h3;
31                  if(count == 12)
32                      begin
33                          count <= 0;
34                          state <= state + 1;
35                      end
36                  else
37                      count<=count+1;
38                  end
39

```


Arduino LCD keypad shield reference document

```
40 2:begin
41 EN<=0;
42 if(count ==205000)
43     begin
44         count <= 0;
45         state <= state+1;
46     end
47 else
48     count<=count+1;
49 end
50
51 3:begin
52 EN<=1;
53 lcd_cmd<=4'h3;
54 if(count==12)
55     begin
56         count<=0;
57         state<=state+1;
58     end
59 else
60     count<=count+1;
61 end
62
63 4:begin
64 EN<=0;
65 if(count == 5000)
66     begin
67         count<=0;
68         state<=state+1;
69     end
70 else
71     count<=count+1;
72 end
73
74 5:begin
75 EN<=1;
76 lcd_cmd<=4'h3;
77 if(count == 12)
78     begin
79         count<=0;
80         state<=state+1;
81     end
82 else
83     count<=count+1;
84 end
85
86 6:begin
87 EN<=0;
88 if(count == 2000)
89     begin
90         count <=0;
91         state<=state+1;
92     end
93 else
94     count<=count+1;
95 end
96
97 7:begin
98 EN<=1;
99 lcd_cmd<=4'h2;
100 if(count == 12)
101     begin
102         count<=0;
103         state<=state+1;
104     end
105 else
106     count<=count+1;
107 end
108
109 8:begin
110 EN<=0;
111 if(count ==2000)
112     begin
113         count<=0;
```

Arduino LCD keypad shield reference document

```
192 15:begin
193 if(count == 12)
194     begin
195         EN<=0;
196         count<=0;
197         state<=state+1;
198     end
199 else
200     count<=count+1;
201 end
202
203 16:begin
204 if(count == 2000)
205     begin
206         count <=0;
207         state<=state+1;
208     end
209 else
210     count<=count+1;
211 end
212
213 17:begin
214 EN<=1;
215 lcd_cmd<=4'h0;
216 if(count == 12)
217     begin
218         count<=0;
219         state<=state+1;
220     end
221 else
222     count<=count+1;
223 end
224
225 18:begin
226 EN<=0;
227 if(count == 50)
228     begin
229         EN<=1;
230         lcd_cmd<=4'h6;
231         count<=0;
232         state<=state+1;
233     end
234 else
235     count<=count+1;
236 end
237
238 19:begin
239 if(count == 12)
240     begin
241         EN<=0;
242         count<=0;
243         state<=state+1;
244     end
245 else
246     count<=count+1;
247 end
248
249 20:begin
250 if(count == 2000)
251     begin
252         count <= 0;
253         state<=state+1;
254     end
255 else
256     count<=count+1;
257 end
258
259 21:begin
260 EN<=1;
261 lcd_cmd<=4'h0;
262 if(count == 12)
263     begin
264         count<=0;
265         state<=state+1;
266     end
267 else
268     count<=count+1;
269 end
270
271 22:begin
272 EN<=0;
```

porda açıklanmıştır..

Arduino LCD keypad shield reference document

```
272 | EN<=0;
273 | if(count == 50)
274 |     begin
275 |         EN<=1;
276 |         lcd_cmd<=4'h1;
277 |         count<=0;
278 |         state<=state+1;
279 |     end
280 | else
281 |     count<=count+1;
282 | end
283 |
284 | 23:begin
285 | if(count == 12)
286 |     begin
287 |         EN<=0;
288 |         count<=0;
289 |         state<=state+1;
290 |     end
291 | else
292 |     count<=count+1;
293 | end
294 |
295 |
296 | 24:begin
297 | if(count == 5000)
298 |     begin
299 |         count <= 0;
300 |         state<=state+1;
301 |     end
302 | else
303 |     count<=count+1;
304 | end
305 |
306 | 25:begin
307 | EN<=1;
308 | lcd_cmd<=4'h8;
309 | if(count == 12)
310 |     begin
311 |         count<=0;
312 |         state<=state+1;
313 |     end
314 | else
315 |     count<=count+1;
316 | end
317 |
318 | 26:begin
319 | EN<=0;
320 | if(count == 50)
321 |     begin
322 |         EN<=1;
323 |         lcd_cmd<=4'h0;
324 |         count<=0;
325 |         state<=state+1;
326 |     end
327 | else
328 |     count<=count+1;
329 | end
330 |
331 | 27:begin
332 | if(count == 30)
333 |     begin
334 |         EN<=0;
335 |         count<=0;
336 |         state<=state+1;
337 |     end
338 | else
339 |     count<=count+1;
340 | end
341 |
342 | 28:begin
343 | if(count == 10000)
344 |     begin
345 |         count<=0;
346 |         state<=state+1;
347 |     end
348 | else
349 |     count<=count+1;
350 | end
```

Arduino LCD keypad shield reference document

```
351 | //-----İnstuaction register'a veri yazımı bitti. Aşağıda data register'a FURKAN yazım kodu vardır.
352 |         29:begin
353 |             EN<=1;
354 |             RS<=1;
355 |             lcd_cmd<=4'h4;
356 |             if(count == 30)
357 |                 begin
358 |                     count <=0;
359 |                     state<=state+1;
360 |                 end
361 |             else
362 |                 count<=count+1;
363 |             end
364 |
365 |         30:begin
366 |             EN<=0;
367 |             if(count == 125)
368 |                 begin
369 |                     EN<=1;
370 |                     lcd_cmd<=4'h6;
371 |                     state<=state+1;
372 |                 end
373 |             else
374 |                 count<=count+1;
375 |             end
376 |
377 |         31:begin
378 |             if(count == 30)
379 |                 begin
380 |                     EN<=0;
381 |                     count<=0;
382 |                     state<=state+1;
383 |                 end
384 |             else
385 |                 count<=count+1;
386 |             end
387 |
388 |         32:begin
389 |             if(count == 5000)
390 |                 begin
391 |                     count<=0;
392 |                     state<=state+1;
393 |                 end
394 |             else
395 |                 count<=count+1;
396 |             end
397 |         //F harfi bitti U'ya geçtik.
398 |         33:begin
399 |             EN<=1;
400 |             RS<=1;
401 |             lcd_cmd<=4'h5;
402 |             if(count == 30)
403 |                 begin
404 |                     count<=0;
405 |                     state<=state+1;
406 |                 end
407 |             else
408 |                 count<=count+1;
409 |             end
410 |
411 |         34:begin
412 |             EN<=0;
413 |             if(count == 125)
414 |                 begin
415 |                     EN<=1;
416 |                     lcd_cmd<=4'h5;
417 |                     count<=0;
418 |                     state<=state+1;
419 |                 end
420 |             else
421 |                 count<=count+1;
422 |             end
423 |
424 |         35:begin
425 |             if(count == 30)
426 |                 begin
427 |                     EN<=0;
428 |                     count<=0;
429 |                     state<=state+1;
430 |                 end
431 |             else
432 |                 count<=count+1;
433 |             end
434 |         end
435 |     end
436 | end
```

Arduino LCD keypad shield reference document

```
430     end
431   else
432     count<=count+1;
433   end
434
435   36:begin
436   if(count == 5000)
437     begin
438       count<=0;
439       state<=state+1;
440     end
441   else
442     count<=count+1;
443   end
444   //U harfi yazıldı. R harfinin yazma işlemi başladı.
445   37:begin
446   EN<=1;
447   RS<=1;
448   lcd_cmd<=4'h5;
449   if(count==30)
450     begin
451       count<=0;
452       state<=state+1;
453     end
454   else
455     count<=count+1;
456   end
457
458   38:begin
459   EN<=0;
460   if(count == 125)
461     begin
462       EN<=1;
463       lcd_cmd<=4'h2;
464       count<=0;
465       state<=state+1;
466     end
467   else
468     count<=count+1;
469   end
470
471   39:begin
472   if(count == 30)
473     begin
474       EN<=0;
475       count<=0;
476       state<=state+1;
477     end
478   else
479     count<=count+1;
480   end
481
482   40:begin
483   if(count == 5000)
484     begin
485       count<=0;
486       state<=state+1;
487     end
488   else
489     count<=count+1;
490   end
491   //R harfi yazıldı K harfinin yazılma işlemi başladı.
492   41:begin
493   EN<=1;
494   RS<=1;
495   lcd_cmd<=4'h4;
496   if(count == 30)
497     begin
498       count <=0;
499       state<=state+1;
500     end
501   else
502     count<=count+1;
503   end
504
505   42:begin
506   EN<=0;
507   if(count == 125)
508     begin
509       EN<=1;
510       lcd_cmd<=4'h8;
511       state<=state+1;
```

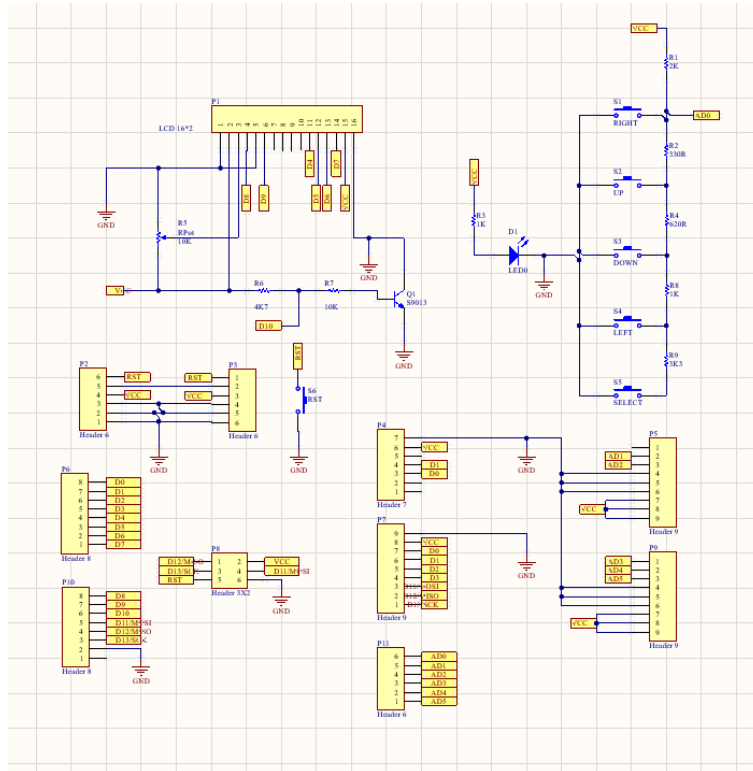
Arduino LCD keypad shield reference document

```
--- |
512 |         end
513 |     else
514 |         count<=count+1;
515 |     end
516 |
517 | 43:begin
518 | if(count == 30)
519 |     begin
520 |         EN<=0;
521 |         count<=0;
522 |         state<=state+1;
523 |     end
524 | else
525 |     count<=count+1;
526 | end
527 |
528 | 44:begin
529 | if(count == 5000)
530 |     begin
531 |         count<=0;
532 |         state<=state+1;
533 |     end
534 | else
535 |     count<=count+1;
536 | end
537 | //K harfi yazıldı. A harfinin yazılma işlemi başladı.
538 | 45:begin
539 | EN<=1;
540 | RS<=1;
541 | lcd_cmd<=4'h4;
542 | if(count == 30)
543 |     begin
544 |         count<=0;
545 |         state<=state+1;
546 |     end
547 | else
548 |     count<=count+1;
549 | end
550 |
551 | 46:begin
552 | EN<=0;
553 | if(count == 125)
554 |     begin
555 |         EN<=1;
556 |         lcd_cmd<=4'h1;
557 |         count<=0;
558 |         state<=state+1;
559 |     end
560 | else
561 |     count<=count+1;
562 | end
563 |
564 | 47:begin
565 | if(count == 30)
566 |     begin
567 |         EN<=0;
568 |         count<=0;
569 |         state<=state+1;
570 |     end
571 | else
572 |     count<=count+1;
573 | end
574 |
575 | 48:begin
576 | if(count == 5000)
577 |     begin
578 |         count<=0;
579 |         state<=state+1;
580 |     end
581 | else
582 |     count<=count+1;
583 | end
584 | //A harfi yazıldı. N harfinin yazılma işlemi başladı.
585 | 49:begin
586 | EN<=1;
587 | RS<=1;
588 | lcd_cmd<=4'h4;
589 | if(count==30)
590 |     begin
591 |         count<=0;
```

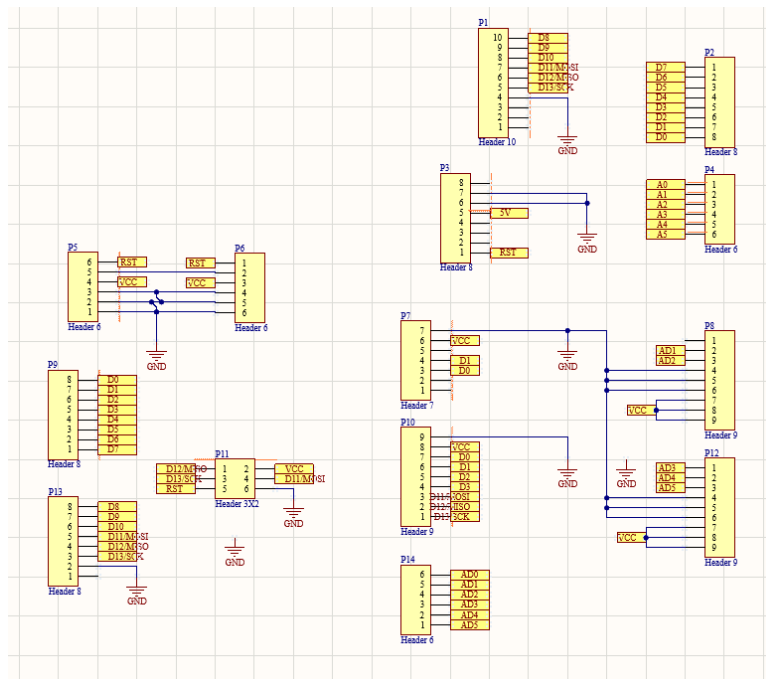
Arduino LCD keypad shield reference document

```
592 |         state<=state+1;
593 |     end
594 | else
595 |     count<=count+1;
596 | end
597 |
598 | 50:begin
599 |     EN<=0;
600 |     if(count == 125)
601 |         begin
602 |             EN<=1;
603 |             lcd_cmd<=4'hE;
604 |             count<=0;
605 |             state<=state+1;
606 |         end
607 |     else
608 |         count<=count+1;
609 |     end
610 |
611 | 51: begin
612 |     if(count == 30)
613 |         begin
614 |             EN<=0;
615 |             count<=0;
616 |             state<=state+1;
617 |         end
618 |     else
619 |         count<=count+1;
620 |     end
621 |
622 | 52:begin
623 |     if(count == 5000)
624 |         begin
625 |             count<=0;
626 |             state<=state+1;
627 |         end
628 |     else
629 |         count<=count+1;
630 |     end
631 |     //Ekрана FURKAN yazıldı.
632 | endcase
633 | end
634 | endmodule
```

4. SCHEMATICS

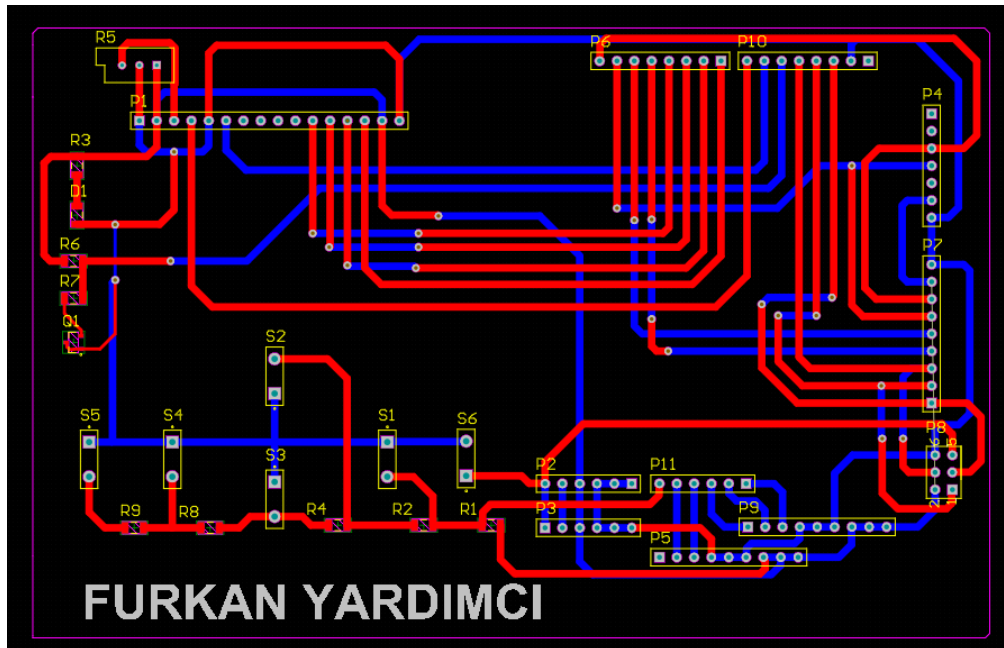


Img 5: Schematic1

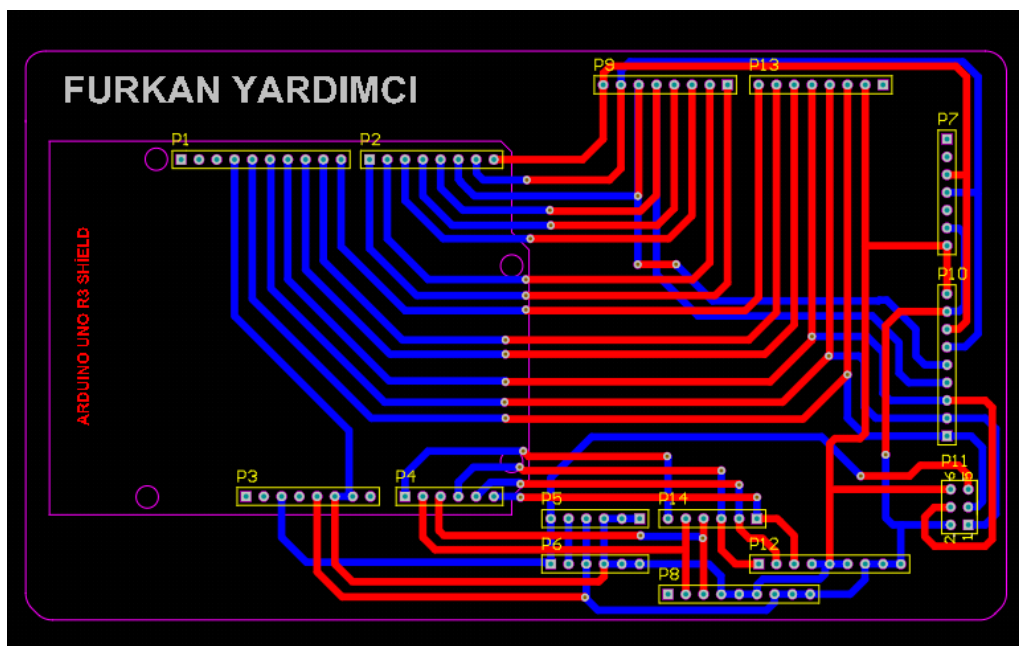


Img 6: Schematic2

5. PCBS

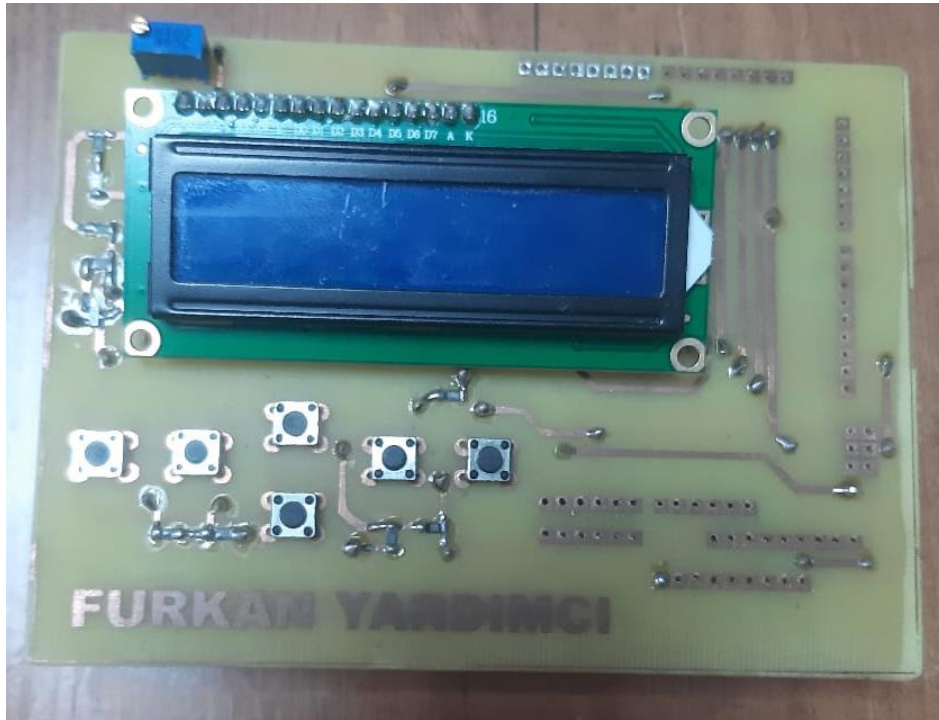


Img 7: PCB1

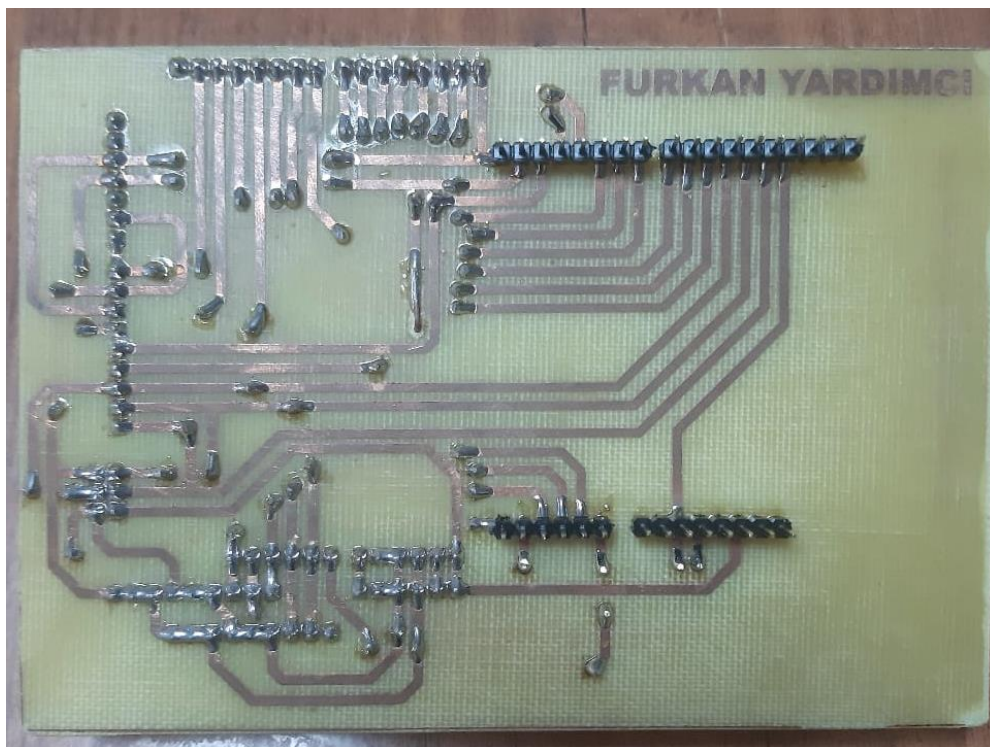


Img 8: PCB2

6. FINAL VIEW OF PROJECT



Img 9: Front view of the project



Img 10: Back view of the project

7.REFERENCES:

- [1] <https://www.youtube.com/watch?v=8YYZVVcnVpM&t=141s>
- [2] <https://linuxhint.com/interface-lcd-4-bit-8-bit-modes-arduino/>
- [3] [ARDUINO LCD KEYPAD SHIELD ÖRNEK UYGULAMA - Elektronik Bilgi Paylaşım Platformu \(arduinocuyuz.blogspot.com\)](http://arduinocuyuz.blogspot.com)
- [4] [PYNQ - Python productivity for Zynq - Board](#)
- [5] [PYNQ-Z2 Setup Guide — Python productivity for Zynq \(Pynq\)](#)
- [6] [A PYNQ-Z2 Guide for Absolute Dummies — Part I: Fun with LEDs and Switches | by Umer Farooq | Medium \(umer-farooq.com\)](#)
- [7] [16x2 LCD Display Module - Pinout & Datasheet \(circuitdigest.com\)](#)