STM32F407 Based 74 Series IC testing system at Gate level

Dron Patel
Department of Electronics and Communication
Institute of Technology
Nirma University, Ahmedabad, Gujarat, India
21bec089@nirmauni.ac.in

Kushal Patel
Department of Electronics and Communication
Institute of Technology
Nirma University, Ahmedabad, Gujarat, India
21bec90@nirmauni.ac.in

***Abstract*—Various ICs come with a different range of configurations. There are plenty of techniques available in the markets for the IC testing which are totally expensive using microcontroller based integrated circuits(IC's). So we produced simple design, accurate and low budget embedded IC tester with the user- friendly effectiveness. Here its comprises of both the hardware and software architectures for this design to implement. In this we are interfacing hardware with the stm32f407 programming in order to know the working condition of an IC. The result is low-cost automatic test equipment, able to execute a preliminary digital test, using just a laptop and an stm32f407 interfacing. Here we implemented the continuity and leakage test with built tester. Programming in the stm32f407 IDE to display the output based on the measured voltage in LCD. Interfacing with the stm32f407 MEGA controller with the program in stm32f407 IDE.**

I. INTRODUCTION

The IC tester simply determines which are workable gates and which are faulty. The main purpose of the project is to develop a digital IC tester that is very less expensive and handy than that of what are available in markets. The aim is to check the ICs in very due course of time and display results of ICs being good or faulty immediately. The necessary input signal conditions are applied to the inputs of the gate through microcontroller and output of each gate is monitored and compared with the truth table, and depending on that comparison IC is tested whether it is good or faulty. The basic function of digital IC tester is to test the logic functioning of the ICs as described in the truth table/function table. The truth tables are stored in database while coding of the microcontroller. The test displays the good ICs and faulty ICs on LCD. The test is being accomplished with the ICs belonging to the basic logic gate IC series. There are many IC tester available in market, varieties of choices are present for users. But we have developed a tester that is very cheap, portable, easy to handle as well as reconfigurable. The paper is organized as follows. Section II defines the problem statement, section III elaborates the proposed system, section IV shows the system design, section V Hardware implementations and Section VI Software implementations gives a future scope for the system developed and then results and conclusion at last in section

II. PROBLEM STATEMENT

In general it is the most important requirement of testing the ICs before using them for IC based work. Faulty ICs gives us results that are inappropriate to work and unexpected. These wrong outputs results leads in wrong evaluation and analysis.

Time and resources are very important in this developing era. And one cannot afford wasting so much of time and resources in finding silly faults caused by faulty ICs. Thus there is a need to develop a low cost and easy maintainable, user friendly digital IC tester.

III. PROPOSED SYSTEM

In our project we are building up a dedicated Economic IC tester. So, performing the linkage and continuity test. This test equipment will be performing both functionality with tested design maintaining without performing any of the reconfigurations, such a sequential operation gives the desired flow by detecting the error in the code. Here we mainly focus on the technical hardware implementation and then interfacing with the software coding and it can be performed different arithmetic circuits like different digital IC testing equipment. Enhance the performance to implement the continuity and leakage tests with the built tester. For enactment, the Programming in the stm32f407 IDE to execute the output by the measured voltage in LCD.

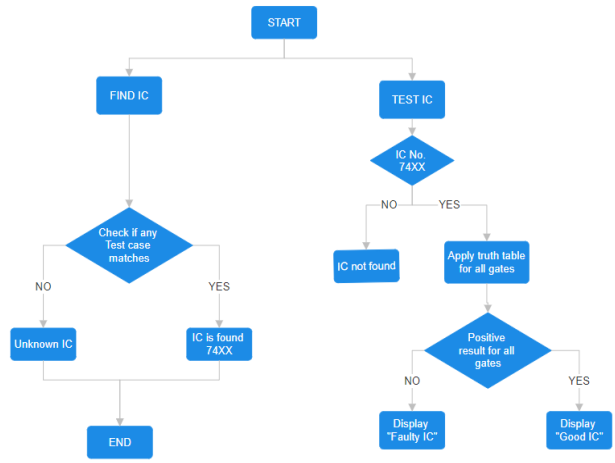


Fig. 1. Flow Diagram

IV. SYSTEM DESIGN

The system's flowchart, which is depicted in Figure 1, essentially explains in detail how the system operates. The algorithm and software were created in accordance with this flowchart of the testing procedure. Additionally, according to this flowchart, the first step is entering the IC No. or four times letter A (AAAA) for unknown ICs, after which the system checks to see if the IC is already in its database. If it is, testing the IC will begin based on the truth table of the ICs gates. Alternatively, if the user enters AAAA for unknown ICs, the system will check the first gate for all ICs in the database, after which the user will select the IC.
The source code for the stm32f407 was created using the flowchart as a guide. The microcontroller will then be programmed with the source code following that. To ensure that the IC tester functions properly and efficiently, testing and calibration will be conducted. If errors are discovered, they will be fixed.

If the user enters the wrong number, the outcome will be displayed on the liquid crystal display (LCD) as (IC not found or IC not work), according to the flowchart. If the user does not know the IC No., the must enter (AAAA), in which case the system will attempt to find the IC No. by applying the truth table on the first gate only for All ICs in the database. Once the system has located the IC No., it will test the IC in question based on the No. that has been discovered.

As shown in Figure 2, the ICs 7400, 7408, 7486, and 7432 have the same pin configuration for inputs and outputs pins with two inputs and one output for each gate, but the IC 7404 pin configuration is completely different as well and has one input and one output for each gate. The system must have the flexibility to reprogram each pin as input or output depending on the pin configuration of the IC under test.

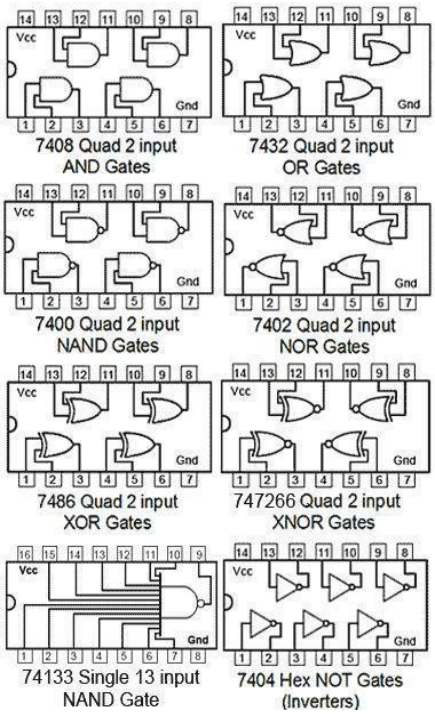
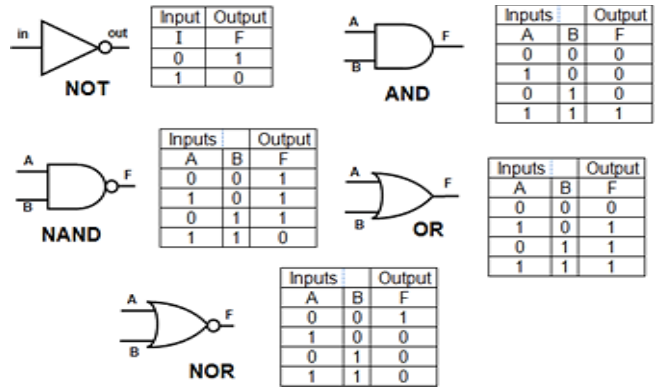


Fig 2. 74-series logic ICs pin diagram (examples)

Therefore, once the user inputs the IC No., the system will reprogram the pins in accordance with the IC No. The system will next apply the truth table of each gate to test whether it operates under appropriate conditions or not. As an example, Figure 3 shows the truth table that the system applies to each gate.



Input	Output
I	F
0	1
1	0

Inputs		Output
A	B	F
0	0	0
1	0	0
0	1	0
1	1	1

Inputs		Output
A	B	F
0	0	1
1	0	1
0	1	1
1	1	0

Inputs		Output
A	B	F
0	0	0
1	0	1
0	1	1
1	1	1

Inputs		Output
A	B	F
0	0	1
1	0	0
0	1	0
1	1	0

Fig 3. Logic gates truth tables (examples)

The system's block diagram is depicted in Figure 4. The system is composed of four main blocks, the first of which is the stm32f407 Mega platform module , which serves as the system's primary microcontroller. The stm32f407 Mega is a cheap platform with capability to connect 54 programmable logic inputs or outputs (from pin 0 to pin 53), and this feature makes it possible to test a variety of logic ICs with larger pin sizes.

The second block represents the IC holder socket; in this instance, the system has 40 pins for the IC holder, all of which are connected to the logic pins, making it simple to connect any IC of any size to the stm32f407 Mega platform and possible to quickly swap out the IC in question to test an alternative IC.

The third block represents the 44 matrix keypad where the IC number can be entered or (AAAA) for an unidentified IC number.

The fourth block is a 16×2 LCD module with an 11C/12C serial converter to display the IC testing results. Depending on the gate level test, the results will appear for each gate as either "OK" or "NOT OK" depending on the truth table for that gate.

The circuit diagram for the IC testing system is shown in Figure 5. This circuit consists of an stm32f407 Mega platform acting as a microcontroller connected to a 40-pin IC holder, a 4×4 keypad for entering the desired IC number for testing, a 16×2 LCD module displaying the results for each gate of the IC, an IC holder, and a battery for providing 9 volts of power. The device's final working prototype is seen in Figure 6 installed on the same box.

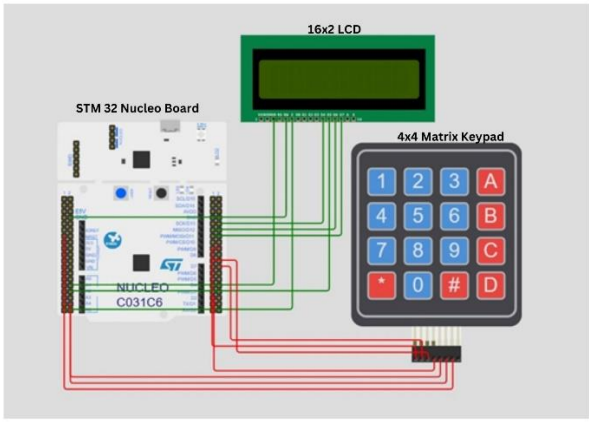


Fig 4 Block diagram of the system

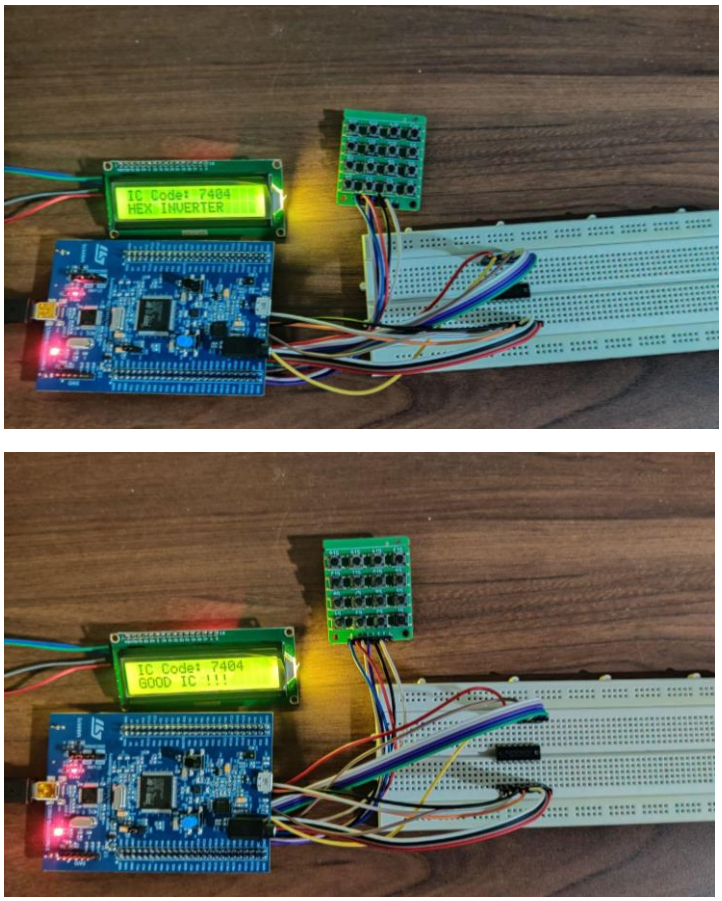


Fig 6 7404 inverter IC working good IC (Hardware implementation)

V. IMPLEMENTATION ON HARDWARE

We can observe the schematic flow of this system process and integrated with the software simulation with the stm32f407. Programming in the stm32f407 IDE to display the output based on the measured voltage in LCD. And, by Interfacing the stm32f407 Mega microcontroller with the program in stm32f407 IDE. By Selecting the pin to check and forcing the current of 100 micro amps to the pin after grounding the Vss and Vdd pins. It measures the voltage across the diode. Based on the measured voltage determining the status of the device.

The LCD Display gets started. The look at system permits you to look to check the IC for common pin opens and for a short-circuit between any pins with one continuity test step. though the network of association information may be manually entered, this information is sometimes self- learned by the System from a known-good part. See the extra data labelled “Continuity Testing for Opens & Short IC.

VI. IMPLEMENTATION ON SOFTWARE

STM32F407 IDE software used for the interference of the Implementation of the various designs of IC testing, Also necessary to have the basic idea of the testing functional and also manually. Hence, we should go throw the flow chart and should develop a source code for the functionality testing. And this project is comprises the part with programing, Hence by that the IC tester should be controls through the programming. If any error condition attained at the time of testing so that we can have the alternate solutions by testing with various methods . In this Implementation stm32f407 operates a major role in execution
And here we used the C programming concepts as it was the basic to write code on stm32f407 platform. After completion of both Hardware and Software part. we have to integrate the whole system with the USB cable so we have to connect the IC tester with PC. Hence can upload the program with the stm32f407.

VII. RESULTS

From the above proposed system, the circuits are interfaced and connected. We have taken samples of various IC’s for the implementation and tested with the leakage test and continuity test. Then the IC testers of both i.e. one is in perfect condition IC, second one defective tester IC could be taken for our analysis of checking which is good working IC and which is not perhaps bad. So from the below interface through the design of logic with various types of IC’s it can be checked and verified. The LCD display as shown in Fig.5 could show the output according to the instructions given. Then the process of the whole system could be shown below. Here we create a different sets of data tables with various IC’s like OR, X-OR, NAND, AND. The program we implemented through the stm32f407 relatively differs the executable values of the output with respectively comparatively with in that 4 gates. If suppose the 1-Gate value if its suits with any of the outcome Gates like AND it executes and displays the gate name in the LCD. If suppose it doesnot matches with any of the outcome values then it shows some error in the Display. In this Scenario entire process could be done similarly.

VIII. FUTURE SCOPE

This project has large potential to be improved by number of ways. This system can be further developed for many other market ICs other than 74 series IC (such has Op-Amp, regulators, comparators etc) used here as it is reconfigurable. The developed IC tester is useful for educational purpose, but further advance development can make it efficient for industrial use. The IC tester project can also be enhanced to be able to identify unmarked devices with the help of auto-

identification mode so that a more powerful and functional IC tester can be provided for users. The wrong/no insertion of the IC can be detected by programming in microcontroller. For example, if an IC is not inserted incorrectly, buzzer will be buzz to inform users with indication on LCD that "IC is inserted incorrectly" or "No IC is inserted"

IX. CONCLUSION

The Digital IC tester tests the ICs being placed in ZIF socket and gives results of ICs being good or faulty. The model is user friendly and is easy to use and maintain. It reduces time to test IC as we are testing it automatically compared to manually testing where various connections and application of inputs is done. The code is reconfigurable. If any further changes are required to make according to the need of user, it can be done very easily. The market price of IC testers is very high whereas this model hardly cost for INR 1500. It saves lots of time and resources and thus is economically efficient.

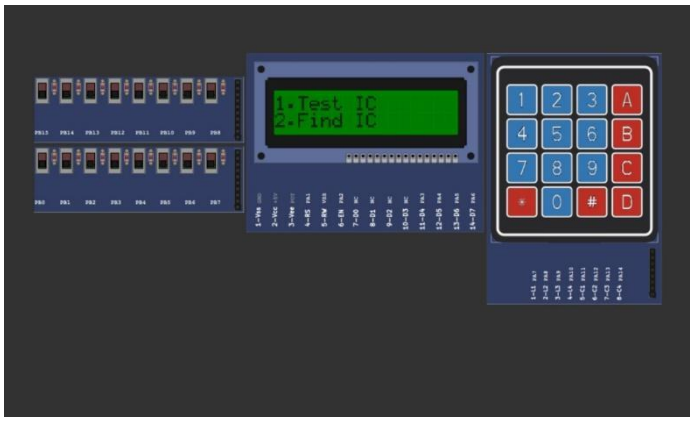


Fig 7 Simulation on PICSim Lab

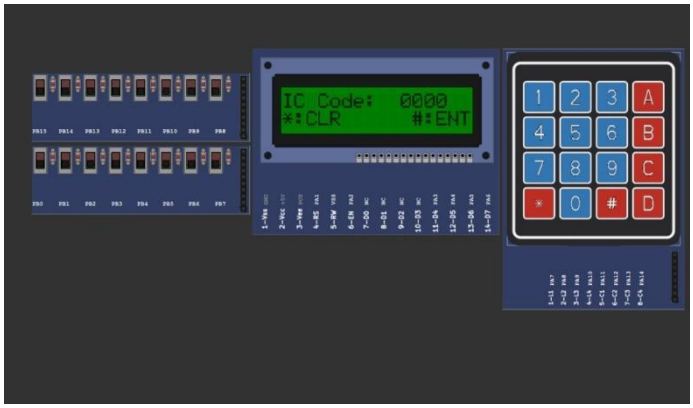


Fig 8 Simulation on PICSim Lab

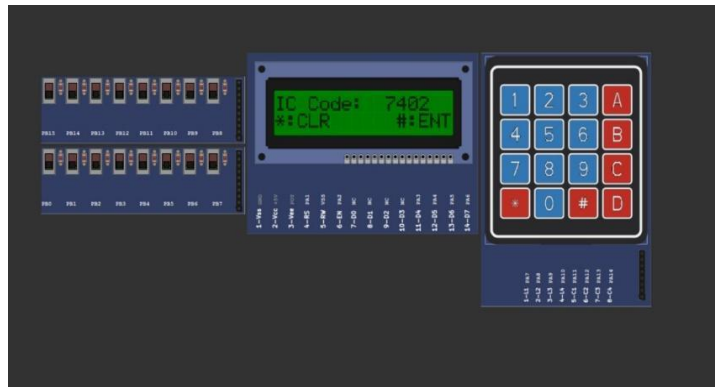


Fig 9 7402 Simulation on PICSim Lab

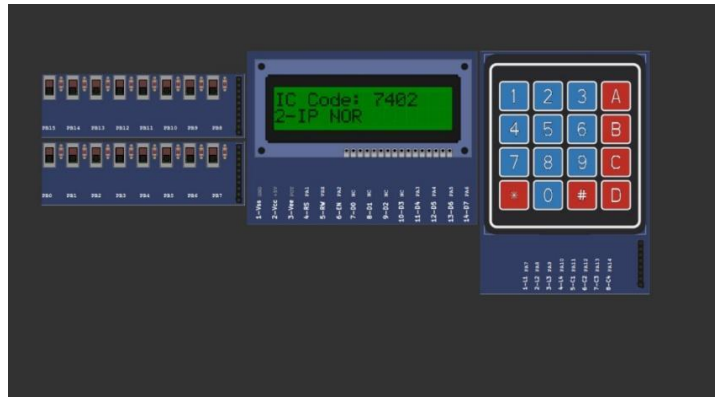


Fig 10 7402 [2-IP NOR] detected

X. REFERENCES

[1] S. P. Lau, G. V Merrett, A. S. Weddell, and N. M. White, "A traffic-aware street lighting scheme for smart cities using autonomous networked sensors," *Computers and Electrical Engineering*, vol. 45, pp. 192–207, Jul. 2015, doi: 10.1016/j.compeleceng.2015.06.011.

[2] K. N. Devika and R. Bhakthavatchalu, "Design of efficient programmable test-per-scan logic BIST modules," in *2017 International conference on Microelectronic Devices, Circuits and Systems (ICMDCS)*, Aug. 2017, pp. 1–6, doi: 10.1109/ICMDCS.2017.8211609.

[3] B. Xiao, X. Li, Y. Lin, and Y. Zhang, "Design and application of an IC chip test instrument based on HT46RU24," in *2021 International Symposium on Computer Technology and Information Science (ISCTIS)*, Jun. 2021, pp. 235–238, doi: 10.1109/ISCTIS51085.2021.00056.

[4] Y.-C. Chang *et al.*, "Design of the multifunction IC-EMC test board with off-board probes for evaluating a microcontroller," in *2015 Asia-Pacific Symposium on Electromagnetic Compatibility (APEMC)*, May 2015, pp. 223–226, doi: 10.1109/APEMC.2015.7175301.

[5] I. J. Hasan, B. M. Waheib, N. A. J. Salih, and N. I. Abdulkhaleq, "A global system for mobile communications-based electrical power consumption for a non-contact smart billing system," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 11, no. 6, pp.

4659–4666, Dec. 2021, doi: 10.11591/ijece.v11i6.pp4659-4666.
[6] Y. Fu, J. Liu, “System design for wearable blood oxygen saturation and pulse measurement device,” 6th International Conference on Applied Human Factors and Ergonomics (AHFE 20 15) and the Affiliated Conferences, Procedia Manuf acturing, vol. 3, pp. 1187–1194, 2015, doi: 10.1016/j.promfg.2015.07.197.
[7] G. C. Seng, S. Salleh, A. R. Harris, M. N. Jamaludd in and I. Kamarulafizam, “Standalone ECG monitor ing system using digital signal processing hardware, ” Proc. of IEEE Conference, January 2012.
[8] M. H. Bhuyan, M. M. Haque, M. A. Rauf and M. M. I. Khan, “Design and Implementation of a Microcontroller B ased Elevator Control Systems,” Proceedings of the International Conference on Engineering Research, Innovation and Education
[9] A. Malvino and D. J. Bates, “Electronic Principles.” 7th edition, Tata McGraw-Hill Publishing Company Limited, New Delhi, Indi a, 2011, pp. 203-223.
[10] S. M. Lee, D. Lee, "Healthcare wearable devices: an analysis of key factors for continuous use intention," Springer Nature, vol. 14, pp. 503– 53115 October 2020.

XI. APPENDIX

PIN TEST.C

```
if (ic_code == 7400 || ic_code == 7403 || ic_code == 7424 || ic_code == 7426 || ic_code == 7437 || ic_code == 7438 ) {

    GPIO_Mode_Change(IC_port,L1, GPIO_MODE_OUTPUT_PP); GPIO_Mode_Change(IC_port,L2, GPIO_MODE_OUTPUT_PP); GPIO_Mode_Change(IC_port,L3, GPIO_MODE_INPUT);
    GPIO_Mode_Change(IC_port,L9, GPIO_MODE_OUTPUT_PP); GPIO_Mode_Change(IC_port,L10, GPIO_MODE_OUTPUT_PP); GPIO_Mode_Change(IC_port,L8, GPIO_MODE_INPUT);
    GPIO_Mode_Change(IC_port,L12, GPIO_MODE_OUTPUT_PP); GPIO_Mode_Change(IC_port,L13, GPIO_MODE_OUTPUT_PP); GPIO_Mode_Change(IC_port,L11, GPIO_MODE_INPUT);
    GPIO_Mode_Change(IC_port,L4, GPIO_MODE_OUTPUT_PP); GPIO_Mode_Change(IC_port,L5, GPIO_MODE_OUTPUT_PP); GPIO_Mode_Change(IC_port,L6, GPIO_MODE_INPUT);

bool check = true;
    HAL_GPIO_WritePin(IC_port,L1, 0); HAL_GPIO_WritePin(IC_port,L2, 0);
    HAL_GPIO_WritePin(IC_port,L4, 0); HAL_GPIO_WritePin(IC_port,L5, 0);
    HAL_GPIO_WritePin(IC_port,L9, 0); HAL_GPIO_WritePin(IC_port,L10, 0);
    HAL_GPIO_WritePin(IC_port,L12, 0); HAL_GPIO_WritePin(IC_port,L13, 0);
    if ( (HAL_GPIO_ReadPin(IC_port,L3) == 1) && (HAL_GPIO_ReadPin(IC_port,L6) == 1) && (HAL_GPIO_ReadPin(IC_port,L8) == 1) && (HAL_GPIO_ReadPin(IC_port,L11) == 1))
      check = check;
    else
      check = false;
    //////////////////////////////////////////
    HAL_GPIO_WritePin(IC_port,L1, 0); HAL_GPIO_WritePin(IC_port,L2, 1);
    HAL_GPIO_WritePin(IC_port,L4, 0); HAL_GPIO_WritePin(IC_port,L5, 1);
    HAL_GPIO_WritePin(IC_port,L9, 0); HAL_GPIO_WritePin(IC_port,L10, 1);
    HAL_GPIO_WritePin(IC_port,L12, 0); HAL_GPIO_WritePin(IC_port,L13, 1);
    if ((HAL_GPIO_ReadPin(IC_port,L3) == 1) && (HAL_GPIO_ReadPin(IC_port,L6) == 1) && (HAL_GPIO_ReadPin(IC_port,L8) == 1) && (HAL_GPIO_ReadPin(IC_port,L11) == 1))
      check = check;
    else
      check = false;
    //////////////////////////////////////////
    HAL_GPIO_WritePin(IC_port,L1, 1); HAL_GPIO_WritePin(IC_port,L2, 0);
    HAL_GPIO_WritePin(IC_port,L4, 1); HAL_GPIO_WritePin(IC_port,L5, 0);
    HAL_GPIO_WritePin(IC_port,L9, 1); HAL_GPIO_WritePin(IC_port,L10, 0);
    HAL_GPIO_WritePin(IC_port,L12, 1); HAL_GPIO_WritePin(IC_port,L13, 0);
    if ((HAL_GPIO_ReadPin(IC_port,L3) == 1) && (HAL_GPIO_ReadPin(IC_port,L6) == 1) && (HAL_GPIO_ReadPin(IC_port,L8) == 1) && (HAL_GPIO_ReadPin(IC_port,L11) == 1))
      check = check;
    else
      check = false;
    //////////////////////////////////////////
    HAL_GPIO_WritePin(IC_port,L1, 1); HAL_GPIO_WritePin(IC_port,L2, 1);
    HAL_GPIO_WritePin(IC_port,L4, 1); HAL_GPIO_WritePin(IC_port,L5, 1);
    HAL_GPIO_WritePin(IC_port,L9, 1); HAL_GPIO_WritePin(IC_port,L10, 1);
    HAL_GPIO_WritePin(IC_port,L12, 1); HAL_GPIO_WritePin(IC_port,L13, 1);
    if ((HAL_GPIO_ReadPin(IC_port,L3) == 0) && (HAL_GPIO_ReadPin(IC_port,L6) == 0) && (HAL_GPIO_ReadPin(IC_port,L8) == 0) && (HAL_GPIO_ReadPin(IC_port,L11) == 0))
      check = check;
    else
      check = false;

    Serial.println(check);
    Serial.println(ic_code);
  }

//////////////////////////////////////
else if (ic_code == 7401) {
```

```
      GPIO_Mode_Change(IC_port,L2,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L3,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L1, GPIO_MODE_INPUT);
      GPIO_Mode_Change(IC_port,L5,
GPIO_MODE_OUTPUT_PP);   GPIO_Mode_Change(IC_port,L6,
GPIO_MODE_OUTPUT_PP);   GPIO_Mode_Change(IC_port,L4,
GPIO_MODE_INPUT);
      GPIO_Mode_Change(IC_port,L8,
GPIO_MODE_OUTPUT_PP);   GPIO_Mode_Change(IC_port,L9,
GPIO_MODE_OUTPUT_PP);  GPIO_Mode_Change(IC_port,L10,
GPIO_MODE_INPUT);
      GPIO_Mode_Change(IC_port,L11,
GPIO_MODE_OUTPUT_PP);  GPIO_Mode_Change(IC_port,L12,
GPIO_MODE_OUTPUT_PP);  GPIO_Mode_Change(IC_port,L13,
GPIO_MODE_INPUT);



      bool check = true;
      HAL_GPIO_WritePin(IC_port,L2,                    0);
HAL_GPIO_WritePin(IC_port,L3, 0);
      HAL_GPIO_WritePin(IC_port,L6,                    0);
HAL_GPIO_WritePin(IC_port,L5, 0);
      HAL_GPIO_WritePin(IC_port,L9,                    0);
HAL_GPIO_WritePin(IC_port,L8, 0);
      HAL_GPIO_WritePin(IC_port,L12,                   0);
HAL_GPIO_WritePin(IC_port,L11, 0);
      if     ((HAL_GPIO_ReadPin(IC_port,L3)    ==    1)    &&
(HAL_GPIO_ReadPin(IC_port,L6)      ==      1)      &&
(HAL_GPIO_ReadPin(IC_port,L10)      ==      1)      &&
(HAL_GPIO_ReadPin(IC_port,L13) == 1))
         check = check;
      else
         check = false;
      ////////////////////////////////////////////////
     HAL_GPIO_WritePin(IC_port,L2,                     0);
HAL_GPIO_WritePin(IC_port,L3, 1);
      HAL_GPIO_WritePin(IC_port,L6,                    0);
HAL_GPIO_WritePin(IC_port,L5, 1);
      HAL_GPIO_WritePin(IC_port,L9,                    0);
HAL_GPIO_WritePin(IC_port,L8, 1);
      HAL_GPIO_WritePin(IC_port,L12,                   0);
HAL_GPIO_WritePin(IC_port,L11, 1);
      if     ((HAL_GPIO_ReadPin(IC_port,L1)    ==    1)    &&
(HAL_GPIO_ReadPin(IC_port,L4)      ==      1)      &&
(HAL_GPIO_ReadPin(IC_port,L10)      ==      1)      &&
(HAL_GPIO_ReadPin(IC_port,L13) == 1))
         check = check;
      else
         check = false;
      ////////////////////////////////////////////
      HAL_GPIO_WritePin(IC_port,L2,                    1);
HAL_GPIO_WritePin(IC_port,L3, 0);
      HAL_GPIO_WritePin(IC_port,L6,                    1);
HAL_GPIO_WritePin(IC_port,L5, 0);
      HAL_GPIO_WritePin(IC_port,L9,                    1);
HAL_GPIO_WritePin(IC_port,L8, 0);
      HAL_GPIO_WritePin(IC_port,L12,1);
HAL_GPIO_WritePin(IC_port,L11,0);
      if     ((HAL_GPIO_ReadPin(IC_port,L1)    ==    1)    &&
(HAL_GPIO_ReadPin(IC_port,L4)      ==      1)      &&
(HAL_GPIO_ReadPin(IC_port,L10)      ==      1)      &&
(HAL_GPIO_ReadPin(IC_port,L13) == 1))
         check = check;
      else
         check = false;
      ////////////////////////////////////////////
      HAL_GPIO_WritePin(IC_port,L2,                    1);
HAL_GPIO_WritePin(IC_port,L3, 1);
      HAL_GPIO_WritePin(IC_port,L6,                    1);
HAL_GPIO_WritePin(IC_port,L5, 1);
      HAL_GPIO_WritePin(IC_port,L9,                    1);
HAL_GPIO_WritePin(IC_port,L8, 1);
      HAL_GPIO_WritePin(IC_port,L12,1);
HAL_GPIO_WritePin(IC_port,L11,1);
      if     ((HAL_GPIO_ReadPin(IC_port,L1)    ==    0)    &&
(HAL_GPIO_ReadPin(IC_port,L4)      ==      0)      &&
(HAL_GPIO_ReadPin(IC_port,L10)      ==      0)      &&
(HAL_GPIO_ReadPin(IC_port,L13) == 0))
         check = check;
      else
         check = false;

      Serial.println(check);
      Serial.println(ic_code);
   }


////////////////////////////////////////////



else if (ic_code == 7402 || ic_code == 7428 || ic_code == 7433)
{

      GPIO_Mode_Change(IC_port,L2,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L3,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L1, GPIO_MODE_INPUT);
      GPIO_Mode_Change(IC_port,L5,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L6,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L4, GPIO_MODE_INPUT);
      GPIO_Mode_Change(IC_port,L8,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L9,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L10, GPIO_MODE_INPUT);
      GPIO_Mode_Change(IC_port,L11,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L12,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L13, GPIO_MODE_INPUT);

      bool check = true;
      HAL_GPIO_WritePin(IC_port,L2,                    0);
HAL_GPIO_WritePin(IC_port,L3, 0);
      HAL_GPIO_WritePin(IC_port,L6,                    0);
HAL_GPIO_WritePin(IC_port,L5, 0);
      HAL_GPIO_WritePin(IC_port,L9,                    0);
HAL_GPIO_WritePin(IC_port,L8, 0);
      HAL_GPIO_WritePin(IC_port,L12,                   0);
HAL_GPIO_WritePin(IC_port,L11, 0);
      if     ((HAL_GPIO_ReadPin(IC_port,L1)    ==    1)    &&
(HAL_GPIO_ReadPin(IC_port,L4)      ==      1)      &&
(HAL_GPIO_ReadPin(IC_port,L10)      ==      1)      &&
(HAL_GPIO_ReadPin(IC_port,L13) == 1))
```

```
      check = check;
    else
      check = false;
    ////////////////////////////////////////////
    HAL_GPIO_WritePin(IC_port,L2,                    0);
HAL_GPIO_WritePin(IC_port,L3, 1);
    HAL_GPIO_WritePin(IC_port,L6,                    0);
HAL_GPIO_WritePin(IC_port,L5, 1);
    HAL_GPIO_WritePin(IC_port,L9,                    0);
HAL_GPIO_WritePin(IC_port,L8, 1);
    HAL_GPIO_WritePin(IC_port,L12,                   0);
HAL_GPIO_WritePin(IC_port,L11, 1);
    if ((HAL_GPIO_ReadPin(IC_port,L1) == 0) &&
(HAL_GPIO_ReadPin(IC_port,L4) == 0) &&
(HAL_GPIO_ReadPin(IC_port,L10) == 0) &&
(HAL_GPIO_ReadPin(IC_port,L13) == 0))
        check = check;
    else
      check = false;
    ////////////////////////////////////////////
    HAL_GPIO_WritePin(IC_port,L2,                    1);
HAL_GPIO_WritePin(IC_port,L3, 0);
    HAL_GPIO_WritePin(IC_port,L6,                    1);
HAL_GPIO_WritePin(IC_port,L5, 0);
    HAL_GPIO_WritePin(IC_port,L9,                    1);
HAL_GPIO_WritePin(IC_port,L8, 0);
    HAL_GPIO_WritePin(IC_port,L12,1);
HAL_GPIO_WritePin(IC_port,L11,0);
    if ((HAL_GPIO_ReadPin(IC_port,L1) == 0) &&
(HAL_GPIO_ReadPin(IC_port,L4) == 0) &&
(HAL_GPIO_ReadPin(IC_port,L10) == 0) &&
(HAL_GPIO_ReadPin(IC_port,L13) == 0))
        check = check;
    else
      check = false;
    ////////////////////////////////////////////
    HAL_GPIO_WritePin(IC_port,L2,                    1);
HAL_GPIO_WritePin(IC_port,L3, 1);
    HAL_GPIO_WritePin(IC_port,L6,                    1);
HAL_GPIO_WritePin(IC_port,L5, 1);
    HAL_GPIO_WritePin(IC_port,L9,                    1);
HAL_GPIO_WritePin(IC_port,L8, 1);
    HAL_GPIO_WritePin(IC_port,L12,1);
HAL_GPIO_WritePin(IC_port,L11,1);
    if ((HAL_GPIO_ReadPin(IC_port,L1) == 0) &&
(HAL_GPIO_ReadPin(IC_port,L4) == 0) &&
(HAL_GPIO_ReadPin(IC_port,L10) == 0) &&
(HAL_GPIO_ReadPin(IC_port,L13) == 0))
        check = check;
    else
      check = false;

    Serial.println(check);
    Serial.println(ic_code);
  }



//////////////////
else if (ic_code == 7404 || ic_code == 7405 || ic_code == 7406 ||
ic_code == 7414 || ic_code == 7416 || ic_code == 7417 || ic_code
== 7419 || ic_code == 7434) {
    bool check = true;

     GPIO_Mode_Change(IC_port,L1,
GPIO_MODE_OUTPUT_PP);  GPIO_Mode_Change(IC_port,L2,
GPIO_MODE_INPUT);
     GPIO_Mode_Change(IC_port,L3,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L4, GPIO_MODE_INPUT);
     GPIO_Mode_Change(IC_port,L5,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L6, GPIO_MODE_INPUT);
     GPIO_Mode_Change(IC_port,L9,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L8, GPIO_MODE_INPUT);
     GPIO_Mode_Change(IC_port,L11,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L10, GPIO_MODE_INPUT);
     GPIO_Mode_Change(IC_port,L13,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L12, GPIO_MODE_INPUT);


    HAL_GPIO_WritePin(IC_port,L1, 0);
    if (HAL_GPIO_ReadPin(IC_port,L2) == 1)
      check = check;
    else
      check = false;

    HAL_GPIO_WritePin(IC_port,L3, 0);
    if (HAL_GPIO_ReadPin(IC_port,L4) == 1)
      check = check;
    else
      check = false;

    HAL_GPIO_WritePin(IC_port,L5, 0);
    if (HAL_GPIO_ReadPin(IC_port,L6) == 1)
      check = check;
    else
      check = false;

    HAL_GPIO_WritePin(IC_port,L9, 0);
    if (HAL_GPIO_ReadPin(IC_port,L8) == 1)
      check = check;
    else
      check = false;

    HAL_GPIO_WritePin(IC_port,L11, 0);
    if (HAL_GPIO_ReadPin(IC_port,L10) == 1)
      check = check;
    else
      check = false;

    HAL_GPIO_WritePin(IC_port,L13, 0);
    if (HAL_GPIO_ReadPin(IC_port,L12) == 1)
      check = check;
    else
      check = false;

    HAL_GPIO_WritePin(IC_port,L1, 1);
    if (HAL_GPIO_ReadPin(IC_port,L2) == 0)
      check = check;
    else
      check = false;

    HAL_GPIO_WritePin(IC_port,L3, 1);
    if (HAL_GPIO_ReadPin(IC_port,L4) == 0)
      check = check;
    else
      check = false;
```

```
    HAL_GPIO_WritePin(IC_port,L5, 1);
    if (HAL_GPIO_ReadPin(IC_port,L6) == 0)
      check = check;
    else
      check = false;

    HAL_GPIO_WritePin(IC_port,L9, 1);
    if (HAL_GPIO_ReadPin(IC_port,L8) == 0)
      check = check;
    else
      check = false;

    HAL_GPIO_WritePin(IC_port,L11, 1);
    if (HAL_GPIO_ReadPin(IC_port,L10) == 0)
      check = check;
    else
      check = false;

    HAL_GPIO_WritePin(IC_port,L13, 1);
    if (HAL_GPIO_ReadPin(IC_port,L12) == 0)
      check = check;
    else
      check = false;

    Serial.println(check);
    Serial.println(ic_code);
  }

////////////////////////////////////////////////


else if (ic_code == 7408 || ic_code == 7409) {

     GPIO_Mode_Change(IC_port,L1,
GPIO_MODE_OUTPUT_PP);  GPIO_Mode_Change(IC_port,L2,
GPIO_MODE_OUTPUT_PP);  GPIO_Mode_Change(IC_port,L3,
GPIO_MODE_INPUT);
     GPIO_Mode_Change(IC_port,L5,
GPIO_MODE_OUTPUT_PP);  GPIO_Mode_Change(IC_port,L4,
GPIO_MODE_OUTPUT_PP);  GPIO_Mode_Change(IC_port,L6,
GPIO_MODE_INPUT);
     GPIO_Mode_Change(IC_port,L10,
GPIO_MODE_OUTPUT_PP);  GPIO_Mode_Change(IC_port,L9,
GPIO_MODE_OUTPUT_PP);  GPIO_Mode_Change(IC_port,L8,
GPIO_MODE_INPUT);
     GPIO_Mode_Change(IC_port,L13,
GPIO_MODE_OUTPUT_PP);  GPIO_Mode_Change(IC_port,L12,
GPIO_MODE_OUTPUT_PP);  GPIO_Mode_Change(IC_port,L11,
GPIO_MODE_INPUT);


    bool check = true;
    HAL_GPIO_WritePin(IC_port,L1,                    0);
HAL_GPIO_WritePin(IC_port,L2, 0);
    HAL_GPIO_WritePin(IC_port,L4,                    0);
HAL_GPIO_WritePin(IC_port,L5, 0);
    HAL_GPIO_WritePin(IC_port,L9,                    0);
HAL_GPIO_WritePin(IC_port,L10, 0);
    HAL_GPIO_WritePin(IC_port,L12,                   0);
HAL_GPIO_WritePin(IC_port,L13, 0);
     if    ((HAL_GPIO_ReadPin(IC_port,L3)    ==    0)    &&
(HAL_GPIO_ReadPin(IC_port,L6)       ==       0)       &&
(HAL_GPIO_ReadPin(IC_port,L8)       ==       0)       &&
(HAL_GPIO_ReadPin(IC_port,L11) == 0))
      check = check;
    else
      check = false;
    //////////////////////////////////////////////
    HAL_GPIO_WritePin(IC_port,L1,                    0);
HAL_GPIO_WritePin(IC_port,L2, 1);
    HAL_GPIO_WritePin(IC_port,L4,                    0);
HAL_GPIO_WritePin(IC_port,L5, 1);
    HAL_GPIO_WritePin(IC_port,L9,                    0);
HAL_GPIO_WritePin(IC_port,L10, 1);
    HAL_GPIO_WritePin(IC_port,L12,                   0);
HAL_GPIO_WritePin(IC_port,L13, 1);
     if    ((HAL_GPIO_ReadPin(IC_port,L3)    ==    0)    &&
(HAL_GPIO_ReadPin(IC_port,L6)       ==       0)       &&
(HAL_GPIO_ReadPin(IC_port,L8)       ==       0)       &&
(HAL_GPIO_ReadPin(IC_port,L11) == 0))
      check = check;
    else
      check = false;
    //////////////////////////////////////////////
    HAL_GPIO_WritePin(IC_port,L1,                    1);
HAL_GPIO_WritePin(IC_port,L2, 0);
    HAL_GPIO_WritePin(IC_port,L4,                    1);
HAL_GPIO_WritePin(IC_port,L5, 0);
    HAL_GPIO_WritePin(IC_port,L9,                    1);
HAL_GPIO_WritePin(IC_port,L10, 0);
    HAL_GPIO_WritePin(IC_port,L12,                   1);
HAL_GPIO_WritePin(IC_port,L13, 0);
     if    ((HAL_GPIO_ReadPin(IC_port,L3)    ==    0)    &&
(HAL_GPIO_ReadPin(IC_port,L6)       ==       0)       &&
(HAL_GPIO_ReadPin(IC_port,L8)       ==       0)       &&
(HAL_GPIO_ReadPin(IC_port,L11) == 0))
      check = check;
    else
      check = false;
    //////////////////////////////////////////////
    HAL_GPIO_WritePin(IC_port,L1,                    1);
HAL_GPIO_WritePin(IC_port,L2, 1);
    HAL_GPIO_WritePin(IC_port,L4,                    1);
HAL_GPIO_WritePin(IC_port,L5, 1);
    HAL_GPIO_WritePin(IC_port,L9,                    1);
HAL_GPIO_WritePin(IC_port,L10, 1);
    HAL_GPIO_WritePin(IC_port,L12,                   1);
HAL_GPIO_WritePin(IC_port,L13, 1);
    if    ((HAL_GPIO_ReadPin(IC_port,L3)    ==    1)    &&
(HAL_GPIO_ReadPin(IC_port,L6)       ==       1)       &&
(HAL_GPIO_ReadPin(IC_port,L8)       ==       1)       &&
(HAL_GPIO_ReadPin(IC_port,L11) == 1))
      check = check;
    else
      check = false;

    Serial.println(check);
    Serial.println(ic_code);
  }
//////////////////////////
// 3ip nand


else if (ic_code == 7410 || ic_code == 7412) {

     GPIO_Mode_Change(IC_port,L1,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L2,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L13,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L12, GPIO_MODE_INPUT);
```

```
    GPIO_Mode_Change(IC_port,L5,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L4,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L3,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L6, GPIO_MODE_INPUT);
    GPIO_Mode_Change(IC_port,L10,
GPIO_MODE_OUTPUT_PP);  GPIO_Mode_Change(IC_port,L9,
GPIO_MODE_OUTPUT_PP); GPIO_Mode_Change(IC_port,L11,
GPIO_MODE_OUTPUT_PP);  GPIO_Mode_Change(IC_port,L8,
GPIO_MODE_INPUT);

    bool check = true;
    HAL_GPIO_WritePin(IC_port,L1,                0);
HAL_GPIO_WritePin(IC_port,L2,                0);
HAL_GPIO_WritePin(IC_port,L13, 0);
    HAL_GPIO_WritePin(IC_port,L4,                0);
HAL_GPIO_WritePin(IC_port,L5,                0);
HAL_GPIO_WritePin(IC_port,L3,  0);
    HAL_GPIO_WritePin(IC_port,L9,                0);
HAL_GPIO_WritePin(IC_port,L10,               0);
HAL_GPIO_WritePin(IC_port,L11, 0);

    if  ((HAL_GPIO_ReadPin(IC_port,L12)  ==  1)  &&
(HAL_GPIO_ReadPin(IC_port,L6)  ==  1)  &&
(HAL_GPIO_ReadPin(IC_port,L8) == 1))
      check = check;
    else
      check = false;
    ////////////////////////////////////////////////
    HAL_GPIO_WritePin(IC_port,L1,                0);
HAL_GPIO_WritePin(IC_port,L2,                0);
HAL_GPIO_WritePin(IC_port,L13, 1);
    HAL_GPIO_WritePin(IC_port,L4,                0);
HAL_GPIO_WritePin(IC_port,L5,                0);
HAL_GPIO_WritePin(IC_port,L3,  1);
    HAL_GPIO_WritePin(IC_port,L9,                0);
HAL_GPIO_WritePin(IC_port,L10,               0);
HAL_GPIO_WritePin(IC_port,L11, 1);

    if  ((HAL_GPIO_ReadPin(IC_port,L12)  ==  1)  &&
(HAL_GPIO_ReadPin(IC_port,L6)  ==  1)  &&
(HAL_GPIO_ReadPin(IC_port,L8) == 1))
      check = check;
    else
      check = false;
    /////
    HAL_GPIO_WritePin(IC_port,L1,                0);
HAL_GPIO_WritePin(IC_port,L2,                1);
HAL_GPIO_WritePin(IC_port,L13, 0);
    HAL_GPIO_WritePin(IC_port,L4,                0);
HAL_GPIO_WritePin(IC_port,L5,                1);
HAL_GPIO_WritePin(IC_port,L3,  0);
    HAL_GPIO_WritePin(IC_port,L9,                0);
HAL_GPIO_WritePin(IC_port,L10,               1);
HAL_GPIO_WritePin(IC_port,L11, 0);

    if  ((HAL_GPIO_ReadPin(IC_port,L12)  ==  1)  &&
(HAL_GPIO_ReadPin(IC_port,L6)  ==  1)  &&
(HAL_GPIO_ReadPin(IC_port,L8) == 1))
      check = check;
    else
      check = false;
    ////
    HAL_GPIO_WritePin(IC_port,L1,                0);
HAL_GPIO_WritePin(IC_port,L2,                1);
HAL_GPIO_WritePin(IC_port,L13, 1);
    HAL_GPIO_WritePin(IC_port,L4,                0);
HAL_GPIO_WritePin(IC_port,L5,                1);
HAL_GPIO_WritePin(IC_port,L3,  1);
    HAL_GPIO_WritePin(IC_port,L9,                0);
HAL_GPIO_WritePin(IC_port,L10,               1);
HAL_GPIO_WritePin(IC_port,L11, 1);

    if  ((HAL_GPIO_ReadPin(IC_port,L12)  ==  1)  &&
(HAL_GPIO_ReadPin(IC_port,L6)  ==  1)  &&
(HAL_GPIO_ReadPin(IC_port,L8) == 1))
      check = check;
    else
      check = false;
    ////////////////////////////////////////////////
    HAL_GPIO_WritePin(IC_port,L1,                1);
HAL_GPIO_WritePin(IC_port,L2,                0);
HAL_GPIO_WritePin(IC_port,L13, 0);
    HAL_GPIO_WritePin(IC_port,L4,                1);
HAL_GPIO_WritePin(IC_port,L5,                0);
HAL_GPIO_WritePin(IC_port,L3,  0);
    HAL_GPIO_WritePin(IC_port,L9,                1);
HAL_GPIO_WritePin(IC_port,L10,               0);
HAL_GPIO_WritePin(IC_port,L11, 0);

    if  ((HAL_GPIO_ReadPin(IC_port,L12)  ==  1)  &&
(HAL_GPIO_ReadPin(IC_port,L6)  ==  1)  &&
(HAL_GPIO_ReadPin(IC_port,L8) == 1))
      check = check;
    else
      check = false;
    /////
    HAL_GPIO_WritePin(IC_port,L1,                1);
HAL_GPIO_WritePin(IC_port,L2,                0);
HAL_GPIO_WritePin(IC_port,L13, 1);
    HAL_GPIO_WritePin(IC_port,L4,                1);
HAL_GPIO_WritePin(IC_port,L5,                0);
HAL_GPIO_WritePin(IC_port,L3,  1);
    HAL_GPIO_WritePin(IC_port,L9,                1);
HAL_GPIO_WritePin(IC_port,L10,               0);
HAL_GPIO_WritePin(IC_port,L11, 1);

    if  ((HAL_GPIO_ReadPin(IC_port,L12)  ==  1)  &&
(HAL_GPIO_ReadPin(IC_port,L6)  ==  1)  &&
(HAL_GPIO_ReadPin(IC_port,L8) == 1))
      check = check;
    else
      check = false;
    ////
     HAL_GPIO_WritePin(IC_port,L1,1);
HAL_GPIO_WritePin(IC_port,L2,                1);
HAL_GPIO_WritePin(IC_port,L13, 0);
    HAL_GPIO_WritePin(IC_port,L4,                1);
HAL_GPIO_WritePin(IC_port,L5,                1);
HAL_GPIO_WritePin(IC_port,L3,  0);
    HAL_GPIO_WritePin(IC_port,L9,                1);
HAL_GPIO_WritePin(IC_port,L10,               1);
HAL_GPIO_WritePin(IC_port,L11, 0);

    if  ((HAL_GPIO_ReadPin(IC_port,L12)  ==  1)  &&
(HAL_GPIO_ReadPin(IC_port,L6)  ==  1)  &&
(HAL_GPIO_ReadPin(IC_port,L8) == 1))
      check = check;
    else
      check = false;
```

```
      /////
       HAL_GPIO_WritePin(IC_port,L1,1);
HAL_GPIO_WritePin(IC_port,L2,                    1);
HAL_GPIO_WritePin(IC_port,L13, 1);
      HAL_GPIO_WritePin(IC_port,L4,              1);
HAL_GPIO_WritePin(IC_port,L5,                    1);
HAL_GPIO_WritePin(IC_port,L3,  1);
      HAL_GPIO_WritePin(IC_port,L9,              1);
HAL_GPIO_WritePin(IC_port,L10,                   1);
HAL_GPIO_WritePin(IC_port,L11, 1);

      if    ((HAL_GPIO_ReadPin(IC_port,L12)    ==    0)    &&
(HAL_GPIO_ReadPin(IC_port,L6)      ==      0)      &&
(HAL_GPIO_ReadPin(IC_port,L8) == 0))
         check = check;
      else
         check = false;

      Serial.println(check);
      Serial.println(ic_code);
   }


//////////////////


else if (ic_code == 7411 || ic_code == 7415) {

       GPIO_Mode_Change(IC_port,L1,
GPIO_MODE_OUTPUT_PP);   GPIO_Mode_Change(IC_port,L2,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L13,  GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L12, GPIO_MODE_INPUT);
       GPIO_Mode_Change(IC_port,L5,
GPIO_MODE_OUTPUT_PP);   GPIO_Mode_Change(IC_port,L4,
GPIO_MODE_OUTPUT_PP);   GPIO_Mode_Change(IC_port,L3,
GPIO_MODE_OUTPUT_PP);   GPIO_Mode_Change(IC_port,L6,
GPIO_MODE_INPUT);
       GPIO_Mode_Change(IC_port,L10,
GPIO_MODE_OUTPUT_PP);   GPIO_Mode_Change(IC_port,L9,
GPIO_MODE_OUTPUT_PP);  GPIO_Mode_Change(IC_port,L11,
GPIO_MODE_OUTPUT_PP);   GPIO_Mode_Change(IC_port,L8,
GPIO_MODE_INPUT);

      bool check = true;
      HAL_GPIO_WritePin(IC_port,L1,              0);
HAL_GPIO_WritePin(IC_port,L2,                    0);
HAL_GPIO_WritePin(IC_port,L13, 0);
      HAL_GPIO_WritePin(IC_port,L4,              0);
HAL_GPIO_WritePin(IC_port,L5,                    0);
HAL_GPIO_WritePin(IC_port,L3,  0);
      HAL_GPIO_WritePin(IC_port,L9,              0);
HAL_GPIO_WritePin(IC_port,L10,                   0);
HAL_GPIO_WritePin(IC_port,L11, 0);

      if    ((HAL_GPIO_ReadPin(IC_port,L12)    ==    0)    &&
(HAL_GPIO_ReadPin(IC_port,L6)      ==      0)      &&
(HAL_GPIO_ReadPin(IC_port,L8) == 0))
         check = check;
      else
         check = false;
      ////////////////////////////////////////////////
      HAL_GPIO_WritePin(IC_port,L1,              0);
HAL_GPIO_WritePin(IC_port,L2,                    0);
HAL_GPIO_WritePin(IC_port,L13, 1);
      HAL_GPIO_WritePin(IC_port,L4,              0);
HAL_GPIO_WritePin(IC_port,L5,                    0);
HAL_GPIO_WritePin(IC_port,L3,  1);
      HAL_GPIO_WritePin(IC_port,L9,              0);
HAL_GPIO_WritePin(IC_port,L10,                   0);
HAL_GPIO_WritePin(IC_port,L11, 1);

      if    ((HAL_GPIO_ReadPin(IC_port,L12)    ==    0)    &&
(HAL_GPIO_ReadPin(IC_port,L6)      ==      0)      &&
(HAL_GPIO_ReadPin(IC_port,L8) == 0))
         check = check;
      else
         check = false;
      /////
      HAL_GPIO_WritePin(IC_port,L1,              0);
HAL_GPIO_WritePin(IC_port,L2,                    1);
HAL_GPIO_WritePin(IC_port,L13, 0);
      HAL_GPIO_WritePin(IC_port,L4,              0);
HAL_GPIO_WritePin(IC_port,L5,                    1);
HAL_GPIO_WritePin(IC_port,L3,  0);
      HAL_GPIO_WritePin(IC_port,L9,              0);
HAL_GPIO_WritePin(IC_port,L10,                   1);
HAL_GPIO_WritePin(IC_port,L11, 0);

      if    ((HAL_GPIO_ReadPin(IC_port,L12)    ==    0)    &&
(HAL_GPIO_ReadPin(IC_port,L6)      ==      0)      &&
(HAL_GPIO_ReadPin(IC_port,L8) == 0))
         check = check;
      else
         check = false;
      ////
      HAL_GPIO_WritePin(IC_port,L1,              0);
HAL_GPIO_WritePin(IC_port,L2,                    1);
HAL_GPIO_WritePin(IC_port,L13, 1);
      HAL_GPIO_WritePin(IC_port,L4,              0);
HAL_GPIO_WritePin(IC_port,L5,                    1);
HAL_GPIO_WritePin(IC_port,L3,  1);
      HAL_GPIO_WritePin(IC_port,L9,              0);
HAL_GPIO_WritePin(IC_port,L10,                   1);
HAL_GPIO_WritePin(IC_port,L11, 1);

      if    ((HAL_GPIO_ReadPin(IC_port,L12)    ==    0)    &&
(HAL_GPIO_ReadPin(IC_port,L6)      ==      0)      &&
(HAL_GPIO_ReadPin(IC_port,L8) == 0))
         check = check;
      else
         check = false;
      ////////////////////////////////////////////////
      HAL_GPIO_WritePin(IC_port,L1,              1);
HAL_GPIO_WritePin(IC_port,L2,                    0);
HAL_GPIO_WritePin(IC_port,L13, 0);
      HAL_GPIO_WritePin(IC_port,L4,              1);
HAL_GPIO_WritePin(IC_port,L5,                    0);
HAL_GPIO_WritePin(IC_port,L3,  0);
      HAL_GPIO_WritePin(IC_port,L9,              1);
HAL_GPIO_WritePin(IC_port,L10,                   0);
HAL_GPIO_WritePin(IC_port,L11, 0);

     if    ((HAL_GPIO_ReadPin(IC_port,L12)    ==    0)    &&
(HAL_GPIO_ReadPin(IC_port,L6)      ==      0)      &&
(HAL_GPIO_ReadPin(IC_port,L8) == 0))
         check = check;
      else
         check = false;
      /////
      HAL_GPIO_WritePin(IC_port,L1,              1);
HAL_GPIO_WritePin(IC_port,L2,                    0);
```

```
HAL_GPIO_WritePin(IC_port,L13, 1);
    HAL_GPIO_WritePin(IC_port,L4,  1);
HAL_GPIO_WritePin(IC_port,L5,  0);
HAL_GPIO_WritePin(IC_port,L3,  1);
    HAL_GPIO_WritePin(IC_port,L9,  1);
HAL_GPIO_WritePin(IC_port,L10,  0);
HAL_GPIO_WritePin(IC_port,L11, 1);

    if ((HAL_GPIO_ReadPin(IC_port,L12) == 0) &&
(HAL_GPIO_ReadPin(IC_port,L6) == 0) &&
(HAL_GPIO_ReadPin(IC_port,L8) == 0))
      check = check;
    else
      check = false;
    ////
     HAL_GPIO_WritePin(IC_port,L1,1);
HAL_GPIO_WritePin(IC_port,L2,  1);
HAL_GPIO_WritePin(IC_port,L13, 0);
    HAL_GPIO_WritePin(IC_port,L4,  1);
HAL_GPIO_WritePin(IC_port,L5,  1);
HAL_GPIO_WritePin(IC_port,L3,  0);
    HAL_GPIO_WritePin(IC_port,L9,  1);
HAL_GPIO_WritePin(IC_port,L10,  1);
HAL_GPIO_WritePin(IC_port,L11, 0);

    if ((HAL_GPIO_ReadPin(IC_port,L12) == 0) &&
(HAL_GPIO_ReadPin(IC_port,L6) == 0) &&
(HAL_GPIO_ReadPin(IC_port,L8) == 0))
      check = check;
    else
      check = false;
    /////
     HAL_GPIO_WritePin(IC_port,L1,1);
HAL_GPIO_WritePin(IC_port,L2,  1);
HAL_GPIO_WritePin(IC_port,L13, 1);
    HAL_GPIO_WritePin(IC_port,L4,  1);
HAL_GPIO_WritePin(IC_port,L5,  1);
HAL_GPIO_WritePin(IC_port,L3,  1);
    HAL_GPIO_WritePin(IC_port,L9,  1);
HAL_GPIO_WritePin(IC_port,L10,  1);
HAL_GPIO_WritePin(IC_port,L11, 1);

    if ((HAL_GPIO_ReadPin(IC_port,L12) == 1) &&
(HAL_GPIO_ReadPin(IC_port,L6) == 1) &&
(HAL_GPIO_ReadPin(IC_port,L8) == 1))
      check = check;
    else
      check = false;

    Serial.println(check);
    Serial.println(ic_code);
  }

////////////////////////////////////////////////////////////////////////////////
// 4 ip NAND

else if (ic_code == 7413 || ic_code == 7418 || ic_code == 7420 ||
ic_code == 7422 || ic_code == 7440) {
    bool check = true;

     GPIO_Mode_Change(IC_port,L1,
GPIO_MODE_OUTPUT_PP);  GPIO_Mode_Change(IC_port,L2,
GPIO_MODE_OUTPUT_PP);  GPIO_Mode_Change(IC_port,L4,
GPIO_MODE_OUTPUT_PP);  GPIO_Mode_Change(IC_port,L5,
GPIO_MODE_OUTPUT_PP);  GPIO_Mode_Change(IC_port,L6,
GPIO_MODE_INPUT);
     GPIO_Mode_Change(IC_port,L10,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L9,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L12,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L13,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L8, GPIO_MODE_INPUT);

    HAL_GPIO_WritePin(IC_port,L1,  0);
HAL_GPIO_WritePin(IC_port,L2,  0);
HAL_GPIO_WritePin(IC_port,L4,  0);
HAL_GPIO_WritePin(IC_port,L5,0);
    HAL_GPIO_WritePin(IC_port,L9,  0);
HAL_GPIO_WritePin(IC_port,L10,  0);
HAL_GPIO_WritePin(IC_port,L12,  0);
HAL_GPIO_WritePin(IC_port,L13,0);

    if ((HAL_GPIO_ReadPin(IC_port,L6) == 1) &&
(HAL_GPIO_ReadPin(IC_port,L8) == 1))
      check = check;
    else
      check = false;
    ////////////////////////////////////////////////

    HAL_GPIO_WritePin(IC_port,L1,  0);
HAL_GPIO_WritePin(IC_port,L2,  0);
HAL_GPIO_WritePin(IC_port,L4,  0);
HAL_GPIO_WritePin(IC_port,L5,1);
    HAL_GPIO_WritePin(IC_port,L9,  0);
HAL_GPIO_WritePin(IC_port,L10,  0);
HAL_GPIO_WritePin(IC_port,L12,  0);
HAL_GPIO_WritePin(IC_port,L13,1);

    if ((HAL_GPIO_ReadPin(IC_port,L6) == 1) &&
(HAL_GPIO_ReadPin(IC_port,L8) == 1))
      check = check;
    else
      check = false;
    ////////////////////////////////////////////////

    HAL_GPIO_WritePin(IC_port,L1,  0);
HAL_GPIO_WritePin(IC_port,L2,  0);
HAL_GPIO_WritePin(IC_port,L4,  1);
HAL_GPIO_WritePin(IC_port,L5,0);
    HAL_GPIO_WritePin(IC_port,L9,  0);
HAL_GPIO_WritePin(IC_port,L10,  0);
HAL_GPIO_WritePin(IC_port,L12,1);
HAL_GPIO_WritePin(IC_port,L13,0);

    if ((HAL_GPIO_ReadPin(IC_port,L6) == 1) &&
(HAL_GPIO_ReadPin(IC_port,L8) == 1))
      check = check;
    else
      check = false;
    ////////////////////////////////////////////////

     HAL_GPIO_WritePin(IC_port,L1,  0);
HAL_GPIO_WritePin(IC_port,L2,0);
HAL_GPIO_WritePin(IC_port,L4,  1);
HAL_GPIO_WritePin(IC_port,L5,1);
    HAL_GPIO_WritePin(IC_port,L9,  0);
HAL_GPIO_WritePin(IC_port,L10,  0);
HAL_GPIO_WritePin(IC_port,L12,1);
HAL_GPIO_WritePin(IC_port,L13,1);
```

```
    if ((HAL_GPIO_ReadPin(IC_port,L6) == 1) &&
(HAL_GPIO_ReadPin(IC_port,L8) == 1))
      check = check;
    else
      check = false;

    ////////////////////////////////////////////
    HAL_GPIO_WritePin(IC_port,L1,      0);
HAL_GPIO_WritePin(IC_port,L2,      1);
HAL_GPIO_WritePin(IC_port,L4,      0);
HAL_GPIO_WritePin(IC_port,L5,0);
    HAL_GPIO_WritePin(IC_port,L9,      0);
HAL_GPIO_WritePin(IC_port,L10,     1);
HAL_GPIO_WritePin(IC_port,L12,0);
HAL_GPIO_WritePin(IC_port,L13,0);

    if ((HAL_GPIO_ReadPin(IC_port,L6) == 1) &&
(HAL_GPIO_ReadPin(IC_port,L8) == 1))
      check = check;
    else
      check = false;

    ////////////////////////////////////////////
    HAL_GPIO_WritePin(IC_port,L1,      0);
HAL_GPIO_WritePin(IC_port,L2,      1);
HAL_GPIO_WritePin(IC_port,L4,      0);
HAL_GPIO_WritePin(IC_port,L5,1);
    HAL_GPIO_WritePin(IC_port,L9,      0);
HAL_GPIO_WritePin(IC_port,L10,     1);
HAL_GPIO_WritePin(IC_port,L12,0);
HAL_GPIO_WritePin(IC_port,L13,1);

    if ((HAL_GPIO_ReadPin(IC_port,L6) == 1) &&
(HAL_GPIO_ReadPin(IC_port,L8) == 1))
      check = check;
    else
      check = false;

    ////////////////////////////////////////////
    HAL_GPIO_WritePin(IC_port,L1,      0);
HAL_GPIO_WritePin(IC_port,L2,      1);
HAL_GPIO_WritePin(IC_port,L4,      1);
HAL_GPIO_WritePin(IC_port,L5,0);
    HAL_GPIO_WritePin(IC_port,L9,      0);
HAL_GPIO_WritePin(IC_port,L10,     1);
HAL_GPIO_WritePin(IC_port,L12,1);
HAL_GPIO_WritePin(IC_port,L13,0);

    if ((HAL_GPIO_ReadPin(IC_port,L6) == 1) &&
(HAL_GPIO_ReadPin(IC_port,L8) == 1))
      check = check;
    else
      check = false;

    ////////////////////////////////////////////

    HAL_GPIO_WritePin(IC_port,L1,      0);
HAL_GPIO_WritePin(IC_port,L2,      1);
HAL_GPIO_WritePin(IC_port,L4,      1);
HAL_GPIO_WritePin(IC_port,L5,1);
    HAL_GPIO_WritePin(IC_port,L9,      0);
HAL_GPIO_WritePin(IC_port,L10,     1);
HAL_GPIO_WritePin(IC_port,L12,1);
HAL_GPIO_WritePin(IC_port,L13,1);

    if ((HAL_GPIO_ReadPin(IC_port,L6) == 1) &&
(HAL_GPIO_ReadPin(IC_port,L8) == 1))
      check = check;
    else
      check = false;
    ////////////////////////

    HAL_GPIO_WritePin(IC_port,L1,      1);
HAL_GPIO_WritePin(IC_port,L2,      0);
HAL_GPIO_WritePin(IC_port,L4,      0);
HAL_GPIO_WritePin(IC_port,L5,0);
    HAL_GPIO_WritePin(IC_port,L9,      1);
HAL_GPIO_WritePin(IC_port,L10,     0);
HAL_GPIO_WritePin(IC_port,L12,     0);
HAL_GPIO_WritePin(IC_port,L13,0);

    if ((HAL_GPIO_ReadPin(IC_port,L6) == 1) &&
(HAL_GPIO_ReadPin(IC_port,L8) == 1))
      check = check;
    else
      check = false;
    ////////////////////////////////////////////

    HAL_GPIO_WritePin(IC_port,L1,      1);
HAL_GPIO_WritePin(IC_port,L2,      0);
HAL_GPIO_WritePin(IC_port,L4,      0);
HAL_GPIO_WritePin(IC_port,L5,1);
    HAL_GPIO_WritePin(IC_port,L9,      1);
HAL_GPIO_WritePin(IC_port,L10,     0);
HAL_GPIO_WritePin(IC_port,L12,0);
HAL_GPIO_WritePin(IC_port,L13,1);

    if ((HAL_GPIO_ReadPin(IC_port,L6) == 1) &&
(HAL_GPIO_ReadPin(IC_port,L8) == 1))
      check = check;
    else
      check = false;
    ////////////////////////////////////////////

    HAL_GPIO_WritePin(IC_port,L1,      1);
HAL_GPIO_WritePin(IC_port,L2,      0);
HAL_GPIO_WritePin(IC_port,L4,      1);
HAL_GPIO_WritePin(IC_port,L5,0);
    HAL_GPIO_WritePin(IC_port,L9,      1);
HAL_GPIO_WritePin(IC_port,L10,     0);
HAL_GPIO_WritePin(IC_port,L12,1);
HAL_GPIO_WritePin(IC_port,L13,0);

    if ((HAL_GPIO_ReadPin(IC_port,L6) == 1) &&
(HAL_GPIO_ReadPin(IC_port,L8) == 1))
      check = check;
    else
      check = false;
    ////////////////////////////////////////////

     HAL_GPIO_WritePin(IC_port,L1,1);
HAL_GPIO_WritePin(IC_port,L2,      0);
HAL_GPIO_WritePin(IC_port,L4,      1);
HAL_GPIO_WritePin(IC_port,L5,1);
    HAL_GPIO_WritePin(IC_port,L9,      1);
HAL_GPIO_WritePin(IC_port,L10,     0);
HAL_GPIO_WritePin(IC_port,L12,1);
HAL_GPIO_WritePin(IC_port,L13,1);

    if ((HAL_GPIO_ReadPin(IC_port,L6) == 1) &&
(HAL_GPIO_ReadPin(IC_port,L8) == 1))
```

```
        check = check;
      else
        check = false;

      ////////////////////////////////////////////
      HAL_GPIO_WritePin(IC_port,L1,                1);
HAL_GPIO_WritePin(IC_port,L2,                1);
HAL_GPIO_WritePin(IC_port,L4,                0);
HAL_GPIO_WritePin(IC_port,L5, 0);
      HAL_GPIO_WritePin(IC_port,L9,                1);
HAL_GPIO_WritePin(IC_port,L10,               1);
HAL_GPIO_WritePin(IC_port,L12,               0);
HAL_GPIO_WritePin(IC_port,L13,0);

      if    ((HAL_GPIO_ReadPin(IC_port,L6)    ==    1)    &&
(HAL_GPIO_ReadPin(IC_port,L8) == 1))
        check = check;
      else
        check = false;

      ////////////////////////////////////////////
      HAL_GPIO_WritePin(IC_port,L1,                1);
HAL_GPIO_WritePin(IC_port,L2,                1);
HAL_GPIO_WritePin(IC_port,L4,                0);
HAL_GPIO_WritePin(IC_port,L5, 1);
      HAL_GPIO_WritePin(IC_port,L9,                1);
HAL_GPIO_WritePin(IC_port,L10,               1);
HAL_GPIO_WritePin(IC_port,L12,               0);
HAL_GPIO_WritePin(IC_port,L13,1);

      if    ((HAL_GPIO_ReadPin(IC_port,L6)    ==    1)    &&
(HAL_GPIO_ReadPin(IC_port,L8) == 1))
        check = check;
      else
        check = false;

      ////////////////////////////////////////////
      HAL_GPIO_WritePin(IC_port,L1,                1);
HAL_GPIO_WritePin(IC_port,L2,                1);
HAL_GPIO_WritePin(IC_port,L4,                1);
HAL_GPIO_WritePin(IC_port,L5,0);
      HAL_GPIO_WritePin(IC_port,L9,                1);
HAL_GPIO_WritePin(IC_port,L10,               1);
HAL_GPIO_WritePin(IC_port,L12,1);
HAL_GPIO_WritePin(IC_port,L13,0);

      if    ((HAL_GPIO_ReadPin(IC_port,L6)    ==    1)    &&
(HAL_GPIO_ReadPin(IC_port,L8) == 1))
        check = check;
      else
        check = false;

      ////////////////////////////////////////////

      HAL_GPIO_WritePin(IC_port,L1,                1);
HAL_GPIO_WritePin(IC_port,L2,                1);
HAL_GPIO_WritePin(IC_port,L4,                1);
HAL_GPIO_WritePin(IC_port,L5,1);
      HAL_GPIO_WritePin(IC_port,L9,                1);
HAL_GPIO_WritePin(IC_port,L10,               1);
HAL_GPIO_WritePin(IC_port,L12,1);
HAL_GPIO_WritePin(IC_port,L13,1);

      if    ((HAL_GPIO_ReadPin(IC_port,L6)    ==    0)    &&
(HAL_GPIO_ReadPin(IC_port,L8) == 0))
        check = check;
      else
        check = false;

      Serial.println(check);
      Serial.println(ic_code);
    }

  ////////////////////////////////////////////



else if (ic_code == 7421) {

      GPIO_Mode_Change(IC_port,L1,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L2,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L4,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L5,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L6, GPIO_MODE_INPUT);
      GPIO_Mode_Change(IC_port,L10,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L9,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L12,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L13,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L8, GPIO_MODE_INPUT);

      bool check = true;

      HAL_GPIO_WritePin(IC_port,L1,                0);
HAL_GPIO_WritePin(IC_port,L2,                0);
HAL_GPIO_WritePin(IC_port,L4,                0);
HAL_GPIO_WritePin(IC_port,L5,0);
      HAL_GPIO_WritePin(IC_port,L9,                0);
HAL_GPIO_WritePin(IC_port,L10,               0);
HAL_GPIO_WritePin(IC_port,L12,               0);
HAL_GPIO_WritePin(IC_port,L13,0);

      if    ((HAL_GPIO_ReadPin(IC_port,L6)    ==    0)    &&
(HAL_GPIO_ReadPin(IC_port,L8) == 0))
        check = check;
      else
        check = false;
      ////////////////////////////////////////////

      HAL_GPIO_WritePin(IC_port,L1,                0);
HAL_GPIO_WritePin(IC_port,L2,                0);
HAL_GPIO_WritePin(IC_port,L4,                0);
HAL_GPIO_WritePin(IC_port,L5,1);
      HAL_GPIO_WritePin(IC_port,L9,                0);
HAL_GPIO_WritePin(IC_port,L10,               0);
HAL_GPIO_WritePin(IC_port,L12,               0);
HAL_GPIO_WritePin(IC_port,L13,1);

      if    ((HAL_GPIO_ReadPin(IC_port,L6)    ==    0)    &&
(HAL_GPIO_ReadPin(IC_port,L8) == 0))
        check = check;
      else
        check = false;
      ////////////////////////////////////////////
```

```
    HAL_GPIO_WritePin(IC_port,L1,    0);
HAL_GPIO_WritePin(IC_port,L2,    0);
HAL_GPIO_WritePin(IC_port,L4,    1);
HAL_GPIO_WritePin(IC_port,L5,0);
    HAL_GPIO_WritePin(IC_port,L9,    0);
HAL_GPIO_WritePin(IC_port,L10,    0);
HAL_GPIO_WritePin(IC_port,L12,1);
HAL_GPIO_WritePin(IC_port,L13,0);

    if  ((HAL_GPIO_ReadPin(IC_port,L6)  ==  0)  &&
(HAL_GPIO_ReadPin(IC_port,L8) == 0))
      check = check;
    else
      check = false;
    ////////////////////////////////////////////////

     HAL_GPIO_WritePin(IC_port,L1,    0);
HAL_GPIO_WritePin(IC_port,L2,0);
HAL_GPIO_WritePin(IC_port,L4,    1);
HAL_GPIO_WritePin(IC_port,L5,1);
    HAL_GPIO_WritePin(IC_port,L9,    0);
HAL_GPIO_WritePin(IC_port,L10,    0);
HAL_GPIO_WritePin(IC_port,L12,1);
HAL_GPIO_WritePin(IC_port,L13,1);

    if  ((HAL_GPIO_ReadPin(IC_port,L6)  ==  0)  &&
(HAL_GPIO_ReadPin(IC_port,L8) == 0))
      check = check;
    else
      check = false;

    ////////////////////////////////////////////////
    HAL_GPIO_WritePin(IC_port,L1,    0);
HAL_GPIO_WritePin(IC_port,L2,    1);
HAL_GPIO_WritePin(IC_port,L4,    0);
HAL_GPIO_WritePin(IC_port,L5,0);
    HAL_GPIO_WritePin(IC_port,L9,    0);
HAL_GPIO_WritePin(IC_port,L10,    1);
HAL_GPIO_WritePin(IC_port,L12,0);
HAL_GPIO_WritePin(IC_port,L13,0);

    if  ((HAL_GPIO_ReadPin(IC_port,L6)  ==  0)  &&
(HAL_GPIO_ReadPin(IC_port,L8) == 0))
      check = check;
    else
      check = false;

    ////////////////////////////////////////////////
    HAL_GPIO_WritePin(IC_port,L1,    0);
HAL_GPIO_WritePin(IC_port,L2,    1);
HAL_GPIO_WritePin(IC_port,L4,    0);
HAL_GPIO_WritePin(IC_port,L5,1);
    HAL_GPIO_WritePin(IC_port,L9,    0);
HAL_GPIO_WritePin(IC_port,L10,    1);
HAL_GPIO_WritePin(IC_port,L12,0);
HAL_GPIO_WritePin(IC_port,L13,1);

    if  ((HAL_GPIO_ReadPin(IC_port,L6)  ==  0)  &&
(HAL_GPIO_ReadPin(IC_port,L8) == 0))
      check = check;
    else
      check = false;

    ////////////////////////////////////////////////
    HAL_GPIO_WritePin(IC_port,L1,    0);
HAL_GPIO_WritePin(IC_port,L2,    1);
HAL_GPIO_WritePin(IC_port,L4,    1);
HAL_GPIO_WritePin(IC_port,L5,0);
    HAL_GPIO_WritePin(IC_port,L9,    0);
HAL_GPIO_WritePin(IC_port,L10,    1);
HAL_GPIO_WritePin(IC_port,L12,1);
HAL_GPIO_WritePin(IC_port,L13,0);

    if  ((HAL_GPIO_ReadPin(IC_port,L6)  ==  0)  &&
(HAL_GPIO_ReadPin(IC_port,L8) == 0))
      check = check;
    else
      check = false;

    ////////////////////////////////////////////////
    HAL_GPIO_WritePin(IC_port,L1,    0);
HAL_GPIO_WritePin(IC_port,L2,    1);
HAL_GPIO_WritePin(IC_port,L4,    1);
HAL_GPIO_WritePin(IC_port,L5,1);
    HAL_GPIO_WritePin(IC_port,L9,    0);
HAL_GPIO_WritePin(IC_port,L10,    1);
HAL_GPIO_WritePin(IC_port,L12,1);
HAL_GPIO_WritePin(IC_port,L13,1);

    if  ((HAL_GPIO_ReadPin(IC_port,L6)  ==  0)  &&
(HAL_GPIO_ReadPin(IC_port,L8) == 0))
      check = check;
    else
      check = false;
////////////////////////

    HAL_GPIO_WritePin(IC_port,L1,    1);
HAL_GPIO_WritePin(IC_port,L2,    0);
HAL_GPIO_WritePin(IC_port,L4,    0);
HAL_GPIO_WritePin(IC_port,L5,0);
    HAL_GPIO_WritePin(IC_port,L9,    1);
HAL_GPIO_WritePin(IC_port,L10,    0);
HAL_GPIO_WritePin(IC_port,L12,    0);
HAL_GPIO_WritePin(IC_port,L13,0);

    if  ((HAL_GPIO_ReadPin(IC_port,L6)  ==  0)  &&
(HAL_GPIO_ReadPin(IC_port,L8) == 0))
      check = check;
    else
      check = false;
    ////////////////////////////////////////////////

    HAL_GPIO_WritePin(IC_port,L1,    1);
HAL_GPIO_WritePin(IC_port,L2,    0);
HAL_GPIO_WritePin(IC_port,L4,    0);
HAL_GPIO_WritePin(IC_port,L5,1);
    HAL_GPIO_WritePin(IC_port,L9,    1);
HAL_GPIO_WritePin(IC_port,L10,    0);
HAL_GPIO_WritePin(IC_port,L12,0);
HAL_GPIO_WritePin(IC_port,L13,1);

    if  ((HAL_GPIO_ReadPin(IC_port,L6)  ==  0)  &&
(HAL_GPIO_ReadPin(IC_port,L8) == 0))
      check = check;
    else
      check = false;
    ////////////////////////////////////////////////

    HAL_GPIO_WritePin(IC_port,L1,    1);
HAL_GPIO_WritePin(IC_port,L2,    0);
HAL_GPIO_WritePin(IC_port,L4,    1);
HAL_GPIO_WritePin(IC_port,L5,0);
    HAL_GPIO_WritePin(IC_port,L9,    0);
HAL_GPIO_WritePin(IC_port,L10,    1);
HAL_GPIO_WritePin(IC_port,L12,1);
HAL_GPIO_WritePin(IC_port,L13,0);

    if  ((HAL_GPIO_ReadPin(IC_port,L6)  ==  0)  &&
(HAL_GPIO_ReadPin(IC_port,L8) == 0))
      check = check;
    else
      check = false;

    ////////////////////////////////////////////////

    HAL_GPIO_WritePin(IC_port,L1,    0);
HAL_GPIO_WritePin(IC_port,L2,    1);
HAL_GPIO_WritePin(IC_port,L4,    1);
HAL_GPIO_WritePin(IC_port,L5,1);
    HAL_GPIO_WritePin(IC_port,L9,    0);
HAL_GPIO_WritePin(IC_port,L10,    1);
HAL_GPIO_WritePin(IC_port,L12,1);
HAL_GPIO_WritePin(IC_port,L13,1);

    if  ((HAL_GPIO_ReadPin(IC_port,L6)  ==  0)  &&
(HAL_GPIO_ReadPin(IC_port,L8) == 0))
      check = check;
    else
      check = false;
    ////////////////////////////////////////////////

    HAL_GPIO_WritePin(IC_port,L1,    1);
HAL_GPIO_WritePin(IC_port,L2,    0);
HAL_GPIO_WritePin(IC_port,L4,    0);
HAL_GPIO_WritePin(IC_port,L5,0);
    HAL_GPIO_WritePin(IC_port,L9,    1);
HAL_GPIO_WritePin(IC_port,L10,    0);
HAL_GPIO_WritePin(IC_port,L12,    0);
HAL_GPIO_WritePin(IC_port,L13,0);

    if  ((HAL_GPIO_ReadPin(IC_port,L6)  ==  0)  &&
(HAL_GPIO_ReadPin(IC_port,L8) == 0))
      check = check;
    else
      check = false;
    ////////////////////////////////////////////////

    HAL_GPIO_WritePin(IC_port,L1,    1);
HAL_GPIO_WritePin(IC_port,L2,    0);
HAL_GPIO_WritePin(IC_port,L4,    0);
HAL_GPIO_WritePin(IC_port,L5,1);
    HAL_GPIO_WritePin(IC_port,L9,    1);
HAL_GPIO_WritePin(IC_port,L10,    0);
HAL_GPIO_WritePin(IC_port,L12,0);
HAL_GPIO_WritePin(IC_port,L13,1);

    if  ((HAL_GPIO_ReadPin(IC_port,L6)  ==  0)  &&
(HAL_GPIO_ReadPin(IC_port,L8) == 0))
      check = check;
    else
      check = false;
    ////////////////////////////////////////////////

    HAL_GPIO_WritePin(IC_port,L1,    1);
HAL_GPIO_WritePin(IC_port,L2,    0);
HAL_GPIO_WritePin(IC_port,L4,    1);
```

```
HAL_GPIO_WritePin(IC_port,L5,0);
      HAL_GPIO_WritePin(IC_port,L9,                1);
HAL_GPIO_WritePin(IC_port,L10,                0);
HAL_GPIO_WritePin(IC_port,L12,1);
HAL_GPIO_WritePin(IC_port,L13,0);

      if     ((HAL_GPIO_ReadPin(IC_port,L6)     ==     0)     &&
(HAL_GPIO_ReadPin(IC_port,L8) == 0))
         check = check;
      else
         check = false;
      ////////////////////////////////////////////////

       HAL_GPIO_WritePin(IC_port,L1,1);
HAL_GPIO_WritePin(IC_port,L2,                0);
HAL_GPIO_WritePin(IC_port,L4,                1);
HAL_GPIO_WritePin(IC_port,L5,1);
      HAL_GPIO_WritePin(IC_port,L9,                1);
HAL_GPIO_WritePin(IC_port,L10,                0);
HAL_GPIO_WritePin(IC_port,L12,1);
HAL_GPIO_WritePin(IC_port,L13,1);

      if     ((HAL_GPIO_ReadPin(IC_port,L6)     ==     0)     &&
(HAL_GPIO_ReadPin(IC_port,L8) == 0))
         check = check;
      else
         check = false;

      ////////////////////////////////////////////////
      HAL_GPIO_WritePin(IC_port,L1,                1);
HAL_GPIO_WritePin(IC_port,L2,                1);
HAL_GPIO_WritePin(IC_port,L4,                0);
HAL_GPIO_WritePin(IC_port,L5, 0);
      HAL_GPIO_WritePin(IC_port,L9,                1);
HAL_GPIO_WritePin(IC_port,L10,                1);
HAL_GPIO_WritePin(IC_port,L12,                0);
HAL_GPIO_WritePin(IC_port,L13,0);

      if     ((HAL_GPIO_ReadPin(IC_port,L6)     ==     0)     &&
(HAL_GPIO_ReadPin(IC_port,L8) == 0))
         check = check;
      else
         check = false;

      ////////////////////////////////////////////////
      HAL_GPIO_WritePin(IC_port,L1,                1);
HAL_GPIO_WritePin(IC_port,L2,                1);
HAL_GPIO_WritePin(IC_port,L4,                0);
HAL_GPIO_WritePin(IC_port,L5, 1);
      HAL_GPIO_WritePin(IC_port,L9,                1);
HAL_GPIO_WritePin(IC_port,L10,                1);
HAL_GPIO_WritePin(IC_port,L12,                0);
HAL_GPIO_WritePin(IC_port,L13,1);

      if     ((HAL_GPIO_ReadPin(IC_port,L6)     ==     0)     &&
(HAL_GPIO_ReadPin(IC_port,L8) == 0))
         check = check;
      else
         check = false;

      ////////////////////////////////////////////////
      HAL_GPIO_WritePin(IC_port,L1,                1);
HAL_GPIO_WritePin(IC_port,L2,                1);
HAL_GPIO_WritePin(IC_port,L4,                1);
HAL_GPIO_WritePin(IC_port,L5,0);
      HAL_GPIO_WritePin(IC_port,L9,                1);
HAL_GPIO_WritePin(IC_port,L10,                1);
HAL_GPIO_WritePin(IC_port,L12,1);
HAL_GPIO_WritePin(IC_port,L13,0);

      if     ((HAL_GPIO_ReadPin(IC_port,L6)     ==     0)     &&
(HAL_GPIO_ReadPin(IC_port,L8) == 0))
         check = check;
      else
         check = false;

      ////////////////////////////////////////////////

      HAL_GPIO_WritePin(IC_port,L1,                1);
HAL_GPIO_WritePin(IC_port,L2,                1);
HAL_GPIO_WritePin(IC_port,L4,                1);
HAL_GPIO_WritePin(IC_port,L5,1);
      HAL_GPIO_WritePin(IC_port,L9,                1);
HAL_GPIO_WritePin(IC_port,L10,                1);
HAL_GPIO_WritePin(IC_port,L12,1);
HAL_GPIO_WritePin(IC_port,L13,1);

      if     ((HAL_GPIO_ReadPin(IC_port,L6)     ==     1)     &&
(HAL_GPIO_ReadPin(IC_port,L8) == 1))
         check = check;
      else
         check = false;

      Serial.println(check);
      Serial.println(ic_code);
   }

   ////////////////////////////

else if (ic_code == 7427)
   {


       GPIO_Mode_Change(IC_port,L1,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L2,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L13,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L12, GPIO_MODE_INPUT);
       GPIO_Mode_Change(IC_port,L5,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L4,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L3,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L6, GPIO_MODE_INPUT);
       GPIO_Mode_Change(IC_port,L10,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L9,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L11,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L8, GPIO_MODE_INPUT);

      bool check = true;
      HAL_GPIO_WritePin(IC_port,L1,                0);
HAL_GPIO_WritePin(IC_port,L2,                0);
HAL_GPIO_WritePin(IC_port,L13, 0);
      HAL_GPIO_WritePin(IC_port,L4,                0);
HAL_GPIO_WritePin(IC_port,L5,                0);
HAL_GPIO_WritePin(IC_port,L3,  0);
```

```
      HAL_GPIO_WritePin(IC_port,L9,   0);
HAL_GPIO_WritePin(IC_port,L10,   0);
HAL_GPIO_WritePin(IC_port,L11, 0);

      if  ((HAL_GPIO_ReadPin(IC_port,L12)  ==  1)  &&
(HAL_GPIO_ReadPin(IC_port,L6)  ==  1)  &&
(HAL_GPIO_ReadPin(IC_port,L8) == 1))
         check = check;
      else
         check = false;
      ////////////////////////////////////////////
      HAL_GPIO_WritePin(IC_port,L1,   0);
HAL_GPIO_WritePin(IC_port,L2,   0);
HAL_GPIO_WritePin(IC_port,L13, 1);
      HAL_GPIO_WritePin(IC_port,L4,   0);
HAL_GPIO_WritePin(IC_port,L5,   0);
HAL_GPIO_WritePin(IC_port,L3,  1);
      HAL_GPIO_WritePin(IC_port,L9,   0);
HAL_GPIO_WritePin(IC_port,L10,   0);
HAL_GPIO_WritePin(IC_port,L11, 1);

      if  ((HAL_GPIO_ReadPin(IC_port,L12)  ==  0)  &&
(HAL_GPIO_ReadPin(IC_port,L6)  ==  0)  &&
(HAL_GPIO_ReadPin(IC_port,L8) == 0))
         check = check;
      else
         check = false;
      /////
      HAL_GPIO_WritePin(IC_port,L1,   0);
HAL_GPIO_WritePin(IC_port,L2,   1);
HAL_GPIO_WritePin(IC_port,L13, 0);
      HAL_GPIO_WritePin(IC_port,L4,   0);
HAL_GPIO_WritePin(IC_port,L5,   1);
HAL_GPIO_WritePin(IC_port,L3,  0);
      HAL_GPIO_WritePin(IC_port,L9,   0);
HAL_GPIO_WritePin(IC_port,L10,   1);
HAL_GPIO_WritePin(IC_port,L11, 0);

      if  ((HAL_GPIO_ReadPin(IC_port,L12)  ==  0)  &&
(HAL_GPIO_ReadPin(IC_port,L6)  ==  0)  &&
(HAL_GPIO_ReadPin(IC_port,L8) == 0))
         check = check;
      else
         check = false;
      ////
      HAL_GPIO_WritePin(IC_port,L1,   0);
HAL_GPIO_WritePin(IC_port,L2,   1);
HAL_GPIO_WritePin(IC_port,L13, 1);
      HAL_GPIO_WritePin(IC_port,L4,   0);
HAL_GPIO_WritePin(IC_port,L5,   1);
HAL_GPIO_WritePin(IC_port,L3,  1);
      HAL_GPIO_WritePin(IC_port,L9,   0);
HAL_GPIO_WritePin(IC_port,L10,   1);
HAL_GPIO_WritePin(IC_port,L11, 1);

      if  ((HAL_GPIO_ReadPin(IC_port,L12)  ==  0)  &&
(HAL_GPIO_ReadPin(IC_port,L6)  ==  0)  &&
(HAL_GPIO_ReadPin(IC_port,L8) == 0))
         check = check;
      else
         check = false;
      ////////////////////////////////////////////
      HAL_GPIO_WritePin(IC_port,L1,   1);
HAL_GPIO_WritePin(IC_port,L2,   0);
HAL_GPIO_WritePin(IC_port,L13, 0);
      HAL_GPIO_WritePin(IC_port,L4,   1);
HAL_GPIO_WritePin(IC_port,L5,   0);
HAL_GPIO_WritePin(IC_port,L3,  0);
      HAL_GPIO_WritePin(IC_port,L9,   1);
HAL_GPIO_WritePin(IC_port,L10,   0);
HAL_GPIO_WritePin(IC_port,L11, 0);

      if  ((HAL_GPIO_ReadPin(IC_port,L12)  ==  0)  &&
(HAL_GPIO_ReadPin(IC_port,L6)  ==  0)  &&
(HAL_GPIO_ReadPin(IC_port,L8) == 0))
         check = check;
      else
         check = false;
      /////
      HAL_GPIO_WritePin(IC_port,L1,   1);
HAL_GPIO_WritePin(IC_port,L2,   0);
HAL_GPIO_WritePin(IC_port,L13, 1);
      HAL_GPIO_WritePin(IC_port,L4,   1);
HAL_GPIO_WritePin(IC_port,L5,   0);
HAL_GPIO_WritePin(IC_port,L3,  1);
      HAL_GPIO_WritePin(IC_port,L9,   1);
HAL_GPIO_WritePin(IC_port,L10,   0);
HAL_GPIO_WritePin(IC_port,L11, 1);

      if  ((HAL_GPIO_ReadPin(IC_port,L12)  ==  0)  &&
(HAL_GPIO_ReadPin(IC_port,L6)  ==  0)  &&
(HAL_GPIO_ReadPin(IC_port,L8) == 0))
         check = check;
      else
         check = false;
      ////
       HAL_GPIO_WritePin(IC_port,L1,1);
HAL_GPIO_WritePin(IC_port,L2,   1);
HAL_GPIO_WritePin(IC_port,L13, 0);
      HAL_GPIO_WritePin(IC_port,L4,   1);
HAL_GPIO_WritePin(IC_port,L5,   1);
HAL_GPIO_WritePin(IC_port,L3,  0);
      HAL_GPIO_WritePin(IC_port,L9,   1);
HAL_GPIO_WritePin(IC_port,L10,   1);
HAL_GPIO_WritePin(IC_port,L11, 0);

      if  ((HAL_GPIO_ReadPin(IC_port,L12)  ==  0)  &&
(HAL_GPIO_ReadPin(IC_port,L6)  ==  0)  &&
(HAL_GPIO_ReadPin(IC_port,L8) == 0))
         check = check;
      else
         check = false;
      /////
       HAL_GPIO_WritePin(IC_port,L1,1);
HAL_GPIO_WritePin(IC_port,L2,   1);
HAL_GPIO_WritePin(IC_port,L13, 1);
      HAL_GPIO_WritePin(IC_port,L4,   1);
HAL_GPIO_WritePin(IC_port,L5,   1);
HAL_GPIO_WritePin(IC_port,L3,  1);
      HAL_GPIO_WritePin(IC_port,L9,   1);
HAL_GPIO_WritePin(IC_port,L10,   1);
HAL_GPIO_WritePin(IC_port,L11, 1);

      if  ((HAL_GPIO_ReadPin(IC_port,L12)  ==  0)  &&
(HAL_GPIO_ReadPin(IC_port,L6)  ==  0)  &&
(HAL_GPIO_ReadPin(IC_port,L8) == 0))
         check = check;
      else
         check = false;

      Serial.println(check);
      Serial.println(ic_code);
```

```
  }



else if (ic_code == 7432) {

      GPIO_Mode_Change(IC_port,L1,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L2,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L3, GPIO_MODE_INPUT);
      GPIO_Mode_Change(IC_port,L5,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L4,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L6, GPIO_MODE_INPUT);
      GPIO_Mode_Change(IC_port,L10,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L9,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L8, GPIO_MODE_INPUT);
      GPIO_Mode_Change(IC_port,L13,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L12,
GPIO_MODE_OUTPUT_PP);
GPIO_Mode_Change(IC_port,L11, GPIO_MODE_INPUT);


      bool check = true;
      HAL_GPIO_WritePin(IC_port,L1,                    0);
HAL_GPIO_WritePin(IC_port,L2, 0);
      HAL_GPIO_WritePin(IC_port,L4,                    0);
HAL_GPIO_WritePin(IC_port,L5, 0);
      HAL_GPIO_WritePin(IC_port,L9,                    0);
HAL_GPIO_WritePin(IC_port,L10, 0);
      HAL_GPIO_WritePin(IC_port,L12,                   0);
HAL_GPIO_WritePin(IC_port,L13, 0);
       if   ((HAL_GPIO_ReadPin(IC_port,L3)   ==   0)   &&
(HAL_GPIO_ReadPin(IC_port,L6)       ==       0)       &&
(HAL_GPIO_ReadPin(IC_port,L8)       ==       0)       &&
(HAL_GPIO_ReadPin(IC_port,L11) == 0))
          check = check;
      else
          check = false;
      ////////////////////////////////////////////////
      HAL_GPIO_WritePin(IC_port,L1,                    0);
HAL_GPIO_WritePin(IC_port,L2, 1);
      HAL_GPIO_WritePin(IC_port,L4,                    0);
HAL_GPIO_WritePin(IC_port,L5, 1);
      HAL_GPIO_WritePin(IC_port,L9,                    0);
HAL_GPIO_WritePin(IC_port,L10, 1);
      HAL_GPIO_WritePin(IC_port,L12,                   0);
HAL_GPIO_WritePin(IC_port,L13, 1);
 if     ((HAL_GPIO_ReadPin(IC_port,L3)     ==      1)     &&
(HAL_GPIO_ReadPin(IC_port,L6)       ==       1)       &&
(HAL_GPIO_ReadPin(IC_port,L8)       ==       1)       &&
(HAL_GPIO_ReadPin(IC_port,L11) == 1))
          check = check;
      else
          check = false;
      ////////////////////////////////////////////////
      HAL_GPIO_WritePin(IC_port,L1,                    1);
HAL_GPIO_WritePin(IC_port,L2, 0);
      HAL_GPIO_WritePin(IC_port,L4,                    1);
HAL_GPIO_WritePin(IC_port,L5, 0);
      HAL_GPIO_WritePin(IC_port,L9,                    1);
HAL_GPIO_WritePin(IC_port,L10, 0);
      HAL_GPIO_WritePin(IC_port,L12,                   1);
HAL_GPIO_WritePin(IC_port,L13, 0);
       if   ((HAL_GPIO_ReadPin(IC_port,L3)   ==   1)   &&
(HAL_GPIO_ReadPin(IC_port,L6)       ==       1)       &&
(HAL_GPIO_ReadPin(IC_port,L8)       ==       1)       &&
(HAL_GPIO_ReadPin(IC_port,L11) == 1))
          check = check;
      else
          check = false;
      ////////////////////////////////////////////////
      HAL_GPIO_WritePin(IC_port,L1,                    1);
HAL_GPIO_WritePin(IC_port,L2, 1);
      HAL_GPIO_WritePin(IC_port,L4,                    1);
HAL_GPIO_WritePin(IC_port,L5, 1);
      HAL_GPIO_WritePin(IC_port,L9,                    1);
HAL_GPIO_WritePin(IC_port,L10, 1);
      HAL_GPIO_WritePin(IC_port,L12,                   1);
HAL_GPIO_WritePin(IC_port,L13, 1);
       if   ((HAL_GPIO_ReadPin(IC_port,L3)   ==   1)   &&
(HAL_GPIO_ReadPin(IC_port,L6)       ==       1)       &&
(HAL_GPIO_ReadPin(IC_port,L8)       ==       1)       &&
(HAL_GPIO_ReadPin(IC_port,L11) == 1))
          check = check;
      else
          check = false;

      Serial.println(check);
      Serial.println(ic_code);
  }
```

MAIN.C

```
/* USER CODE BEGIN Header */
/**

******************************************************************************
  * @file           : main.c
  * @brief          : Main program body

******************************************************************************
  * @attention
  *
  * Copyright (c) 2024 STMicroelectronics.
  * All rights reserved.
  *
  * This software is licensed under terms that can be found in the LICENSE file
  * in the root directory of this software component.
  * If no LICENSE file comes with this software, it is provided AS-IS.
  *

******************************************************************************
  */
/* USER CODE END Header */
/* Includes ------------------------------------------------------------------*/
#include "main.h"
```

```
/* Private includes ----------------------------------------------------------*/
/* USER CODE BEGIN Includes */
#include "LCD_header.h"
#include "matrix_keypad.h"
#include "bench.h"
#include "dynamic_GPIO.h"
//#include "matrix_keypad.h"
/* USER CODE END Includes */

/* Private typedef -----------------------------------------------------------*/
/* USER CODE BEGIN PTD */

/* USER CODE END PTD */

/* Private define ------------------------------------------------------------*/
/* USER CODE BEGIN PD */

/* USER CODE END PD */

/* Private macro -------------------------------------------------------------*/
/* USER CODE BEGIN PM */

/* USER CODE END PM */

/* Private variables ---------------------------------------------------------*/
I2C_HandleTypeDef hi2c1;

/* USER CODE BEGIN PV */
int operation_flag,reset_flag,ic_code_val,check;
static unsigned int code;
unsigned char flag=0;       // global variables
/* USER CODE END PV */

/* Private function prototypes -----------------------------------------------*/
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
static void MX_I2C1_Init(void);
/* USER CODE BEGIN PFP */

/* USER CODE END PFP */

/* Private user code ---------------------------------------------------------*/
/* USER CODE BEGIN 0 */

/* USER CODE END 0 */

/**
  * @brief  The application entry point.
  * @retval int
  */
int main(void)
{
  /* USER CODE BEGIN 1 */

  /* USER CODE END 1 */

  /* MCU Configuration--------------------------------------------------------*/
  /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
  HAL_Init();

  /* USER CODE BEGIN Init */


  /* USER CODE END Init */

  /* Configure the system clock */
  SystemClock_Config();

  /* USER CODE BEGIN SysInit */

  /* USER CODE END SysInit */

  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  MX_I2C1_Init();
  /* USER CODE BEGIN 2 */
  init_display_controller();
  init_matrix_keypad();
  operation_flag = POWER_ON_SCREEN;
  unsigned char key;
  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
  key = read_matrix_keypad(STATE);
  if (operation_flag == HOME_SCREEN) {
        if (key == 1){
                operation_flag = TEST_SCREEN;
                reset_flag = MODE_RESET;
                clear_screen();
        }
        if (key == 2){
                operation_flag = FIND_SCREEN;
                clear_screen();
                }
  }


  switch (operation_flag){
  case POWER_ON_SCREEN:
       power_on_screen();
       clear_screen();
       operation_flag = HOME_SCREEN;
       break;
  case HOME_SCREEN:
     home_screen();
     break;
  case  TEST_SCREEN:
       if (flag==0){
                ic_code_val = ic_code_screen(key, reset_flag);
                if(flag==1){
                        ic_code_val = reset_flag = MODE_RESET;
                        continue;
                }
       }
       else if(flag==1){
                ic_code_val = ic_code_screen(key, reset_flag);//////////////////////////
       }
```

```
      break;
  case FIND_SCREEN:
      find_screen();
      break;
  case CNFM_SCREEN:
      confirm_screen(key, ic_code_val);
  }
  reset_flag = RESET_NOTHING;

   /* USER CODE END WHILE */

   /* USER CODE BEGIN 3 */
  }
  /* USER CODE END 3 */
}

/**
  * @brief System Clock Configuration
  * @retval None
  */
void SystemClock_Config(void)
{
  RCC_OscInitTypeDef RCC_OscInitStruct = {0};
  RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};

  /** Initializes the RCC Oscillators according to the specified
parameters
  * in the RCC_OscInitTypeDef structure.
  */
  RCC_OscInitStruct.OscillatorType                =
RCC_OSCILLATORTYPE_HSI;
  RCC_OscInitStruct.HSIState = RCC_HSI_ON;
  RCC_OscInitStruct.HSICalibrationValue           =
RCC_HSICALIBRATION_DEFAULT;
  RCC_OscInitStruct.PLL.PLLState = RCC_PLL_NONE;
  if      (HAL_RCC_OscConfig(&RCC_OscInitStruct)      !=
HAL_OK)
  {
    Error_Handler();
  }

  /** Initializes the CPU, AHB and APB buses clocks
  */
  RCC_ClkInitStruct.ClockType                     =
RCC_CLOCKTYPE_HCLK|RCC_CLOCKTYPE_SYSCLK

|RCC_CLOCKTYPE_PCLK1|RCC_CLOCKTYPE_PCLK2;
  RCC_ClkInitStruct.SYSCLKSource                  =
RCC_SYSCLKSOURCE_HSI;
  RCC_ClkInitStruct.AHBCLKDivider                 =
RCC_SYSCLK_DIV1;
  RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV1;
  RCC_ClkInitStruct.APB2CLKDivider = RCC_HCLK_DIV1;

  if              (HAL_RCC_ClockConfig(&RCC_ClkInitStruct,
FLASH_LATENCY_0) != HAL_OK)
  {
    Error_Handler();
  }
}

static void MX_I2C1_Init(void)
{

  /* USER CODE BEGIN I2C1_Init 0 */

  /* USER CODE END I2C1_Init 0 */

  /* USER CODE BEGIN I2C1_Init 1 */

  /* USER CODE END I2C1_Init 1 */
  hi2c1.Instance = I2C1;
  hi2c1.Init.ClockSpeed = 100000;
  hi2c1.Init.DutyCycle = I2C_DUTYCYCLE_2;
  hi2c1.Init.OwnAddress1 = 0;
  hi2c1.Init.AddressingMode                       =
I2C_ADDRESSINGMODE_7BIT;
  hi2c1.Init.DualAddressMode                      =
I2C_DUALADDRESS_DISABLE;
  hi2c1.Init.OwnAddress2 = 0;
  hi2c1.Init.GeneralCallMode                      =
I2C_GENERALCALL_DISABLE;
  hi2c1.Init.NoStretchMode = I2C_NOSTRETCH_DISABLE;
  if (HAL_I2C_Init(&hi2c1) != HAL_OK)
  {
    Error_Handler();
  }
  /* USER CODE BEGIN I2C1_Init 2 */

  /* USER CODE END I2C1_Init 2 */

}

/**
  * @brief GPIO Initialization Function
  * @param None
  * @retval None
  */
static void MX_GPIO_Init(void)
{
  GPIO_InitTypeDef GPIO_InitStruct = {0};
/* USER CODE BEGIN MX_GPIO_Init_1 */
/* USER CODE END MX_GPIO_Init_1 */

  /* GPIO Ports Clock Enable */
  __HAL_RCC_GPIOD_CLK_ENABLE();
  __HAL_RCC_GPIOA_CLK_ENABLE();
  __HAL_RCC_GPIOB_CLK_ENABLE();

  /*Configure GPIO pin Output Level */
  HAL_GPIO_WritePin(GPIOA,
GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3|GPIO_PIN_4

|GPIO_PIN_5|GPIO_PIN_6|GPIO_PIN_7|GPIO_PIN_8
                    |GPIO_PIN_9|GPIO_PIN_10,
GPIO_PIN_RESET);

  /*Configure GPIO pins : PA1 PA2 PA3 PA4
                         PA5 PA6 PA7 PA8
                         PA9 PA10 */
  GPIO_InitStruct.Pin                             =
GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3|GPIO_PIN_4

|GPIO_PIN_5|GPIO_PIN_6|GPIO_PIN_7|GPIO_PIN_8
                    |GPIO_PIN_9|GPIO_PIN_10;
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
  HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);

  /*Configure GPIO pins : PA11 PA12 PA13 */
  GPIO_InitStruct.Pin                             =
```

```
GPIO_PIN_11|GPIO_PIN_12|GPIO_PIN_13;
  GPIO_InitStruct.Mode = GPIO_MODE_INPUT;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);

/* USER CODE BEGIN MX_GPIO_Init_2 */
/* USER CODE END MX_GPIO_Init_2 */
}

/* USER CODE BEGIN 4 */
void clear_screen(void){
lcd_write(CLEAR_DISP_SCREEN,INST_MODE);
HAL_Delay(2);
}

void power_on_screen(void){
unsigned int i;
lcd_string("  Powering ON   ",line1(0));
HAL_Delay(10);
for (i=0;i<16;i++){
      lcd_putch(0xFF,line2(i));     //for printing bar symbol -->
0xff
      HAL_Delay(10);
}
}
void home_screen(void){
lcd_string("1.Test IC       ",line1(0));
lcd_string("2.Find IC       ",line2(0));
}

int ic_code_screen(unsigned char key,int reset_flag){
static unsigned char key_count;

if (reset_flag == MODE_RESET){
      key_count = 0;
      key = ALL_RELEASED;
      code =0;
      lcd_string("IC Code: ",line1(0));
      lcd_string("*:CLR      #:ENT",line2(0));
}
   if ((key != '*') && (key != '#') && (key !=
ALL_RELEASED)) {
       key_count++;
       if (key_count <=4) {
                code = code * 10 + key;
       }
}
else if (key == '*') { // to clear sec or min
      key_count = 0;
       code = 0;
}

lcd_putch((char)(code / 1000) + '0', line1(10));
   lcd_putch((char)((code / 100) % 10) + '0', line1(11));
lcd_putch((char)((code / 10) % 10) + '0', line1(12));
lcd_putch((char)(code % 10) + '0', line1(13));

if (key == '#' && key_count==4) {
      clear_screen();
      operation_flag = CNFM_SCREEN;
      return code;
}
return 0;
}

void confirm_screen(unsigned char key,int ic_code_val){
lcd_string("IC Code: ",line1(0));
//lcd_string("Name: ",line2(0));

lcd_putch((char)(ic_code_val / 1000) + '0', line1(9));
lcd_putch((char)((ic_code_val / 100) % 10) + '0', line1(10));
lcd_putch((char)((ic_code_val / 10) % 10) + '0', line1(11));
lcd_putch((char)(ic_code_val % 10) + '0', line1(12));

const char *str_name;
str_name = test_bench(ic_code_val) ;
lcd_string(str_name,line2(0));
if(key == '#' ){
      clear_screen();
      lcd_string("Confirmed",line1(0));
}
else if(key=='*'){
      clear_screen();
      operation_flag = HOME_SCREEN;
}
}

void find_screen(void){
int find_ic,i,j;
find_ic=7400;
for(i=0;i<50;i++){
      lcd_string("IC Code: ",line1(0));
      lcd_putch((char)((find_ic+i) / 1000) + '0', line1(9));
      lcd_putch((char)(((find_ic+i) / 100) % 10) + '0',
line1(10));
      lcd_putch((char)(((find_ic+i) / 10) % 10) + '0', line1(11));
      lcd_putch((char)((find_ic+i) % 10) + '0', line1(12));
      //test_bench(find_ic+i);

      for (j=0;j<16;j++){
      lcd_putch(0xFF,line2(j));     //for printing bar symbol -->
0xff
      HAL_Delay(1);
      }
      lcd_string("                ",line2(0));
}
lcd_string("Cann't Find IC",line1(0));
HAL_Delay(1500);
clear_screen();
operation_flag = HOME_SCREEN;

}

/* USER CODE END 4 */

/**
  * @brief  This function is executed in case of error
occurrence.
  * @retval None
  */
void Error_Handler(void)
{
  /* USER CODE BEGIN Error_Handler_Debug */
  /* User can add his own implementation to report the HAL
error return state */
  __disable_irq();
  while (1)
  {
  }
  /* USER CODE END Error_Handler_Debug */
}
```

```
#ifdef  USE_FULL_ASSERT
/**
  * @brief  Reports the name of the source file and the source line number
  *         where the assert_param error has occurred.
  * @param  file: pointer to the source file name
  * @param  line: assert_param error line source number
  * @retval None
  */
void assert_failed(uint8_t *file, uint32_t line)
{
  /* USER CODE BEGIN 6 */
  /* User can add his own implementation to report the file name and line number,
     ex: printf("Wrong parameters value: file %s on line %d\r\n", file, line) */
  /* USER CODE END 6 */
}
#endif /* USE_FULL_ASSERT */
```

MATRIX KEYPAD.C

```
#include "stm32f1xx_hal.h"
#include "matrix_keypad.h"
void init_matrix_keypad(void){
/* Making all the Rows HIGH to start with */
HAL_GPIO_WritePin(keypad_port,ROW1,HI);
HAL_GPIO_WritePin(keypad_port,ROW2,HI);
HAL_GPIO_WritePin(keypad_port,ROW3,HI);
HAL_GPIO_WritePin(keypad_port,ROW4,HI);
}

static unsigned char scan_keypad(void)
{


HAL_GPIO_WritePin(keypad_port,ROW1,LOW);
HAL_GPIO_WritePin(keypad_port,ROW2,HI);
HAL_GPIO_WritePin(keypad_port,ROW3,HI);
HAL_GPIO_WritePin(keypad_port,ROW4,HI);
HAL_Delay(10);
   if (HAL_GPIO_ReadPin(keypad_port,COL1) == LOW)
   {
      return 1;
   }
   else  if  (HAL_GPIO_ReadPin(keypad_port,COL2)  ==
LOW)
   {
      return 2;
   }
   else  if  (HAL_GPIO_ReadPin(keypad_port,COL3)  ==
LOW)
   {
      return 3;
   }


HAL_GPIO_WritePin(keypad_port,ROW1,HI);
HAL_GPIO_WritePin(keypad_port,ROW2,LOW);
HAL_GPIO_WritePin(keypad_port,ROW3,HI);
HAL_GPIO_WritePin(keypad_port,ROW4,HI);
HAL_Delay(10);
   if (HAL_GPIO_ReadPin(keypad_port,COL1) == LOW)
   {
      return 4;
   }
   else  if  (HAL_GPIO_ReadPin(keypad_port,COL2)  ==
LOW)
   {
      return 5;
   }
   else  if  (HAL_GPIO_ReadPin(keypad_port,COL3)  ==
LOW)
   {
      return 6;
   }

HAL_GPIO_WritePin(keypad_port,ROW1,HI);
HAL_GPIO_WritePin(keypad_port,ROW2,HI);
HAL_GPIO_WritePin(keypad_port,ROW3,LOW);
HAL_GPIO_WritePin(keypad_port,ROW4,HI);
HAL_Delay(10);
   if (HAL_GPIO_ReadPin(keypad_port,COL1) == LOW)
   {
      return 7;
   }
   else  if  (HAL_GPIO_ReadPin(keypad_port,COL2)  ==
LOW)
   {
      return 8;
   }
   else  if  (HAL_GPIO_ReadPin(keypad_port,COL3)  ==
LOW)
   {
      return 9;
   }


HAL_GPIO_WritePin(keypad_port,ROW1,HI);
HAL_GPIO_WritePin(keypad_port,ROW2,HI);
HAL_GPIO_WritePin(keypad_port,ROW3,HI);
HAL_GPIO_WritePin(keypad_port,ROW4,LOW);
HAL_Delay(10);
   if (HAL_GPIO_ReadPin(keypad_port,COL1) == LOW)
   {
      return '';  ///11 for ''
   }
   else  if  (HAL_GPIO_ReadPin(keypad_port,COL2)  ==
LOW)
   {
      return 0;
   }
   else  if  (HAL_GPIO_ReadPin(keypad_port,COL3)  ==
LOW)
   {
      return '#';   //12 for '#'
   }


   return ALL_RELEASED;
}

unsigned char read_matrix_keypad(unsigned char mode) // LEVEL STATE
{
   static unsigned char once = 1;
   unsigned char key;
```

```
   key = scan_keypad(); // 1 2 3 4  5 6 7 8 9 0 '*' '#' 'A' 'B' 'C'
'D'

   if (mode == LEVEL)
   {
      return key; //// 1 2 3 4  5 6 7 8 9 0 '*' '#' 'A' 'B' 'C' 'D'
ALL_RELEASED
   }
   else // state change
   {
      if ((key != ALL_RELEASED) && once)
      {
         once = 0;
          return key; // 1 2 3 4  5 6 7 8 9 0 '*' '#' 'A' 'B' 'C' 'D'
ALL_RELEASED
      }

      else if (key == ALL_RELEASED)
      {
         once = 1;
      }
   }

   return ALL_RELEASED;
}
```

LCD.C

```
#include "stm32f1xx_hal.h"
#include "LCD_header.h"
extern I2C_HandleTypeDef hi2c1;
#define SLAVE_ADDRESS_LCD 0x4E

void lcd_write(unsigned char byte,unsigned char mode){
char uppernibble,lowernibble;
uppernibble = ((byte>>4)&0x0F);  //extracting upper and
lower nibble
lowernibble = ((byte)&0x0F);

HAL_GPIO_WritePin(LCD_port, RS, mode);   //mode bit for
data/command

HAL_GPIO_WritePin(LCD_port,
D7,((uppernibble>>3)&0x01));
HAL_GPIO_WritePin(LCD_port,
D6,((uppernibble>>2)&0x01));
HAL_GPIO_WritePin(LCD_port,
D5,((uppernibble>>1)&0x01));
HAL_GPIO_WritePin(LCD_port,
D4,((uppernibble>>0)&0x01));  // sending uppernibble
HAL_GPIO_WritePin(LCD_port, E,1);
HAL_Delay(5);
HAL_GPIO_WritePin(LCD_port, E,0);
HAL_Delay(5);


HAL_GPIO_WritePin(LCD_port,
D7,((lowernibble>>3)&0x01));
HAL_GPIO_WritePin(LCD_port,
D6,((lowernibble>>2)&0x01));
HAL_GPIO_WritePin(LCD_port,
D5,((lowernibble>>1)&0x01));  // sending lowernibble
HAL_GPIO_WritePin(LCD_port,
D4,((lowernibble>>0)&0x01));
HAL_GPIO_WritePin(LCD_port, E,1);
HAL_Delay(5);
HAL_GPIO_WritePin(LCD_port, E,0);
HAL_Delay(5);


if (mode == INST_MODE)
{
     uint16_t command_word = 0;
     command_word = byte;  // rs=0
     uint8_t cmd_t[2];
     cmd_t[0] = (command_word>>8)&0xFF;
     cmd_t[1] = (command_word)&0xFF;
     HAL_I2C_Master_Transmit (&hi2c1,
SLAVE_ADDRESS_LCD,(uint8_t *) cmd_t, 2, 100);
}
else if (mode == DATA_MODE){
     uint16_t data_word = 0;
     data_word = 0x4000|byte;  //rs=1
     uint8_t data_t[2];
     data_t[0] = (data_word>>8)&0xFF;
     data_t[1] = (data_word)&0xFF;
     HAL_I2C_Master_Transmit (&hi2c1,
SLAVE_ADDRESS_LCD,(uint8_t *) data_t, 2, 100);
}



}


void init_display_controller(void)
{
   /* Startup Time for the CLCD controller */
HAL_Delay(10);
   /* The CLCD Startup Sequence */
   lcd_write(0x02,INST_MODE);
   lcd_write(0x28,INST_MODE);
 //set of command to initialize lcd
     lcd_write(0x0C,INST_MODE);
     lcd_write(0x06,INST_MODE);
     lcd_write(0x01,INST_MODE);
     lcd_write(0x80,INST_MODE);
}


void lcd_putch(const unsigned char data, unsigned char addr)
{
   lcd_write(addr, INST_MODE);
   lcd_write(data, DATA_MODE);
}


void lcd_string(const char *str, unsigned char addr)
{
   lcd_write(addr, INST_MODE);
   while (*str != '\0')
   {
      lcd_write(*str, DATA_MODE);
      str++;
   }
}
```