

LAB 4: MATRIX KEYPAD INTERFACE USING PROGRAM-CONTROLLED I/O

THE SMK 10-KEY MATRIX KEYPAD

The purpose of this lab is to control a peripheral device with the MC9S12C32/128 microcontroller, interfacing the device to the system through parallel I/O ports and accessing it using program-controlled I/O. The peripheral device for this lab is a matrix keypad, as used on a variety of products (phones, keyless entry systems, appliances, etc.) The keypad is pictured in Figure 1. Note that there are 7 pins on the back of the keypad, with pin spacing (“pitch”) that will allow the pins to be inserted into a standard breadboard.

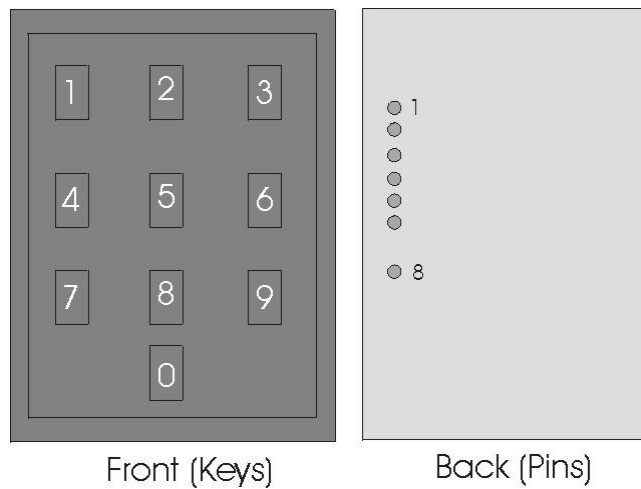


Figure 1. SMK 10-Key Matrix Keypad

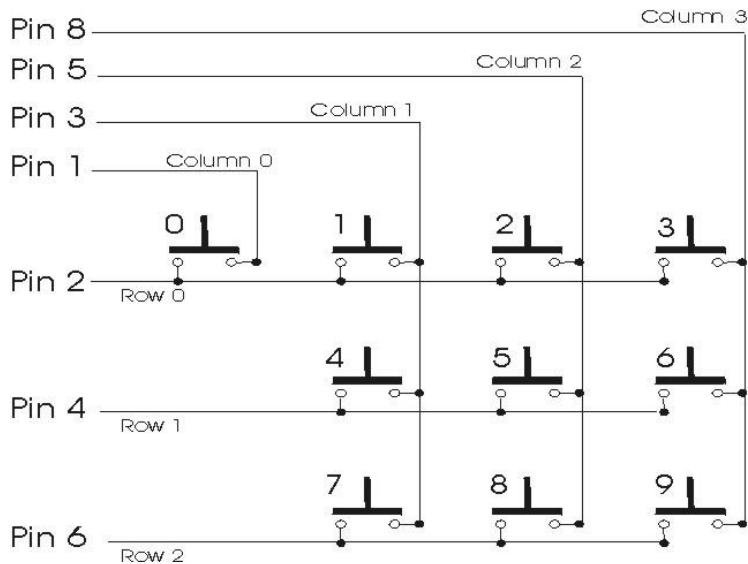


Figure 2. Matrix Keypad Circuit Diagram

The keypad's electrical circuit, shown in Figure 2, contains no active electrical components. It is simply a matrix of wires and switch contacts, arranged in rows and columns, with a key and a pair of contacts at each row-column intersection. Springs normally hold the keys away from the contacts, creating open circuits between the row and column wires. When a key is pressed, the contact closure connects the corresponding row and column wires, creating a short circuit between them. When the key is released, its spring pulls it away from the contacts, restoring the open. Detection of a pressed key thus requires detection of a short circuit between a row wire and a column wire, with the key number determined from the row and column numbers.

INTERFACING THE MATRIX KEYPAD TO THE MICROCONTROLLER

To detect a pressed key and identify the key number, the computer must “scan” the keypad to look for a short circuit condition between a row and a column wire. If a short is detected, the computer must determine which row and column wires are connected to identify the pressed key. The most common way to scan a matrix keypad is to hold all row and column wires at some voltage, $V1$, and then drive one of the column wires C_i to a different voltage, $V2$. If C_i is shorted to one of the row wires, R_k , then C_i will force R_k to voltage $V2$. Otherwise, R_k will remain at voltage $V1$. (Note that the roles of the row and column wires can be reversed if desired.)

To allow the keypad to be scanned as described above, a parallel output port must be used to drive the column lines, and a parallel input port to check the states of the row lines. In this configuration, the voltages used are $V1 = +5$ volts (logic 1) and $V2 = 0$ volts (logic 0). Each of the three row lines should be connected to +5 volts through a pull-up resistor of about 10K ohms to pull the line up to the logic 1 state. The output port should be used drive the column lines to desired states. When not scanning the keypad, the column lines would normally be driven to their “inactive” (logic 1) states. To scan the keypad, one of the column lines should be “activated”, i.e. driven to the logic 0 state, with the other column lines driven to their inactive (logic 1) states. Since the row lines are pulled up to logic 1 through pull-up resistors, forcing one of them to the logic 0 state can be done by shorting it to an activated column line (a column line being driven to the logic 0 state). Therefore, when reading the states of the row lines via the input port, if the state of any row line is logic 0, it must be shorted to the activated column line. Note that if a row line is shorted to an inactive column line (in the logic 1 state), then the row line will remain in the logic 1 state.

The keypad scanning process requires activating one column, with the other three deactivated, and sampling the row lines. If any row is in the logic 0 state, then a key is being pressed at the intersection of that row and the activated column. If the rows are all high, then no key is being pressed in the activated column. This process is repeated in a continuous loop, activating each column, with the others deactivated, until a pressed key is detected.

PRE-LAB ASSIGNMENT

Reading

Review section 18.2 of the Cady text (2nd edition) or 11-1 & Example 11-17 of the Brey text, which discuss simple input devices, including an example of a 16-key matrix keypad.

Hardware Design

This lab will again utilize the Wytec “DragonFly12” 40-pin DIP module. For reference the pin numbering is shown in Figures 3 and 4 at the end of this document. The full DragonFly12 schematic diagram is linked to the course web page. The breadboard stations have banana jacks for connecting the power supply, a 50-pin header for connecting the logic analyzer, and a variety of switches, LEDs, 7-segment display digits and decoders for debugging. The 7-segment display digits include binary to 7-segment decoders, to which connections are made via the headers below the digits.

Parallel input and output ports T and AD are to be used for the keypad interface. The only additional hardware required is the keypad and the 10K pull-up resistors for the keypad row lines. The I/O port connections to the “peripheral devices” should be made as listed in Table 1. Note that Port T drives two “devices” – the keypad column lines and the 7-segment display decoder. Port AD provides data from the keypad row lines.

Parallel Port Pins	Connected Devices
Port T, bits 0-3	7-segment display decoder
Port T, bits 7-4	Keypad column lines (Columns 0-3)
Port AD, bits 0-2	Keypad row lines (Rows 0-2)

Table 1. Parallel input/output port connections.

To aid in debugging, be prepared to use the logic analyzer and/or oscilloscope to observe the states of the various peripheral device lines.

Software Design

Review the earlier labs to recall how to initialize and access I/O ports in C, and to set/clear/test individual bits of a byte.

Your test program should execute in a continuous loop, scanning the keypad for pressed keys. If a pressed key is detected, the key number should be displayed on a 7-segment display digit. Refer to the discussion above for a description of the keypad scanning process. Note that when writing to the output port to change the column lines, the display should remain unchanged, and when changing the display, the column lines should remain unchanged. (Hint – keep a copy of the last value written to the output port, modify the bits to be changed, and then write the changed value to the output port.) Likewise, when reading the row lines, the bits corresponding to the switches must be “masked”.

Thoroughly comment your program to demonstrate your understanding of printer operation.

LAB PROCEDURE

1. Mount the DragonFly12 module on your breadboard.
2. Connect the power supply pins (+5v and ground) on the module to the corresponding pins on the breadboard station.
3. Connect the Port T pins to the 7-segment display digit.
4. Mount the keypad on your breadboard, and connect it to Ports T and AD, utilizing 10K resistors (provided) as pullups.
5. After double-checking connections, connect the power supply to the breadboard station.
6. Verify that can access the I/O ports properly, using the test methods learned in previous labs. If you experience problems, you should test the I/O ports using the test programs from Lab 2 and the logic analyzer.
7. Enter, compile and download your keypad scanning program. Execute your program, verifying that correct key number is displayed for each of the 10 keys.
8. Demonstrate your working programs to the lab instructor.

LAB REPORT

1. Briefly describe your hardware design. Include a schematic diagram.
2. Include a printout of your C program, including well thought through comments.
3. Discuss your results.

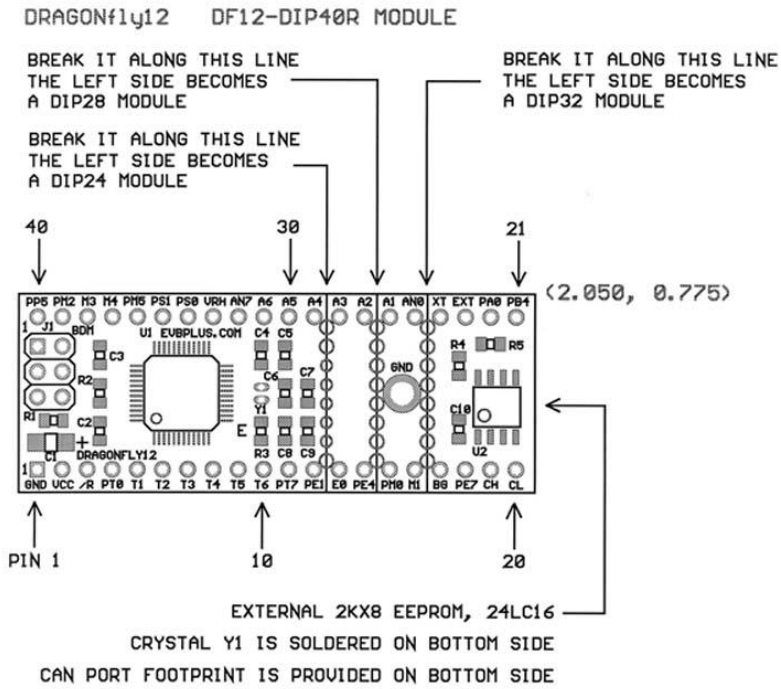


Figure 3. Wytec DragonFly12 module layout (top)
http://www.evbplus.com/c32_modules/c32.html

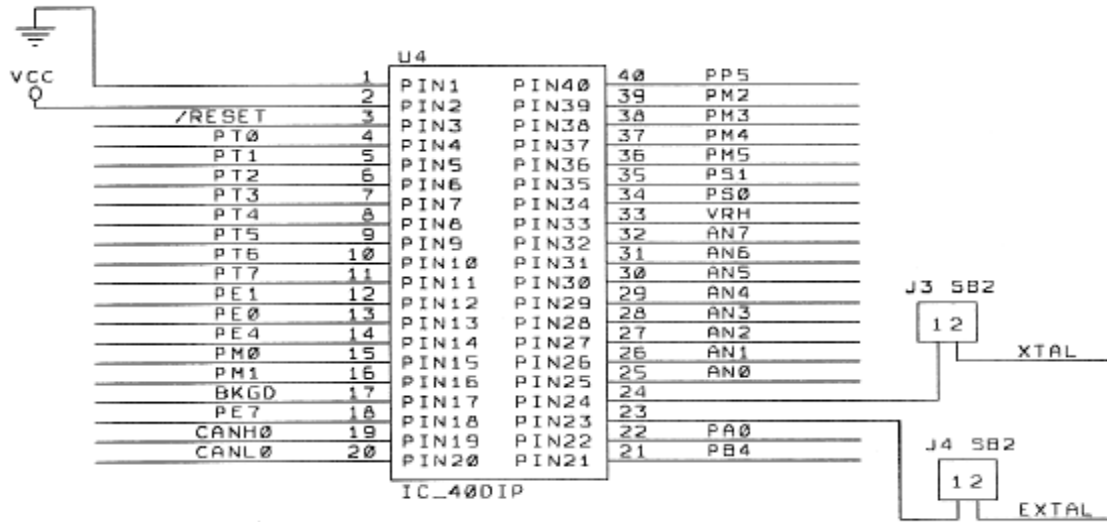


Figure 4. Wytec DragonFly12 40-pin DIP MC9S12C128 signal connections.
http://www.evbplus.com/c32_modules/c32.html