The information below was kindly written up for EmuAdvice by Marshall, based on his experiences when building his arcade controls. You can also download this document as a printer-friendly Word .doc file by clicking here or an Adobe .pdf file by clicking here. Additionally, Marshall provided us with a quick explanation on how to get around the "Type "OK" Screen" when MAME starts a game if you're using your controller and don't have any keys at hand, which is now explained on this page. Thanks Marshall!!

HOW TO HACK A KEYBOARD FOR USE IN MAME

UPDATE (23Oct2003):

Added a graphic by tmasman showing how to wire the switches to the keyboard hack.

UPDATE (23Sep2003):

Somehow in the first edition of this page, I didn't see that a keyboard hack can also work for 3-player 3-button games and 4-player 2-button games as well as the standard Street Fighter 2-player, 6-button layout. Details and a brief run-through of how to make this work are now included. Also, the first two Appendices have been revised to show 4-player key assignments.

Also, later versions of MAME (probably from 0.60 up) now allow use of the Windows "Flag" keys on the keyboard, but you need some tricks to make them work. Details added.

Gamepad hacks were left out of the alternatives section of the page. Details added. Also completely revised the alternatives section as well.

Added a paragraph on "build quality" to the keyboard hacks disadvantages section.

BTW, I no longer recommend keyboard hacks. I am glad I wrote the page, I will continue to maintain it, and I am glad it has helped people, but with products like the KeyWiz Eco offering 32 non-ghosting inputs, Shazaaam! (shift-key) functionality, better response time, and re-programmability for under $30, it is hard to justify. I still think a keyboard hack will do what I say, and if you are on a limited budget, only need a few controls, and only want to play MAME (or other apps with assignable inputs), it makes sense, but for the majority of users, a simple encoder is more desirable in terms of cost/time involved/functionality. (Added to Procedure also).

Added some keys to avoid (direction arrows, among others) to the main procedure. Information initially provided by RandyT (KeyWiz). This had an unexpected positive in that by avoiding the arrow keys, I was able to map J1 Up and Down to KP 8 and KP 2 and the Player 1 buttons to MAME defaults on most of the keyboards, making them more useful for scrolling front-end menus by using the HotRod support options.

Clarified the "Idealized, Advanced concepts" section.

Revised all appendices for consistency and to highlight some errors.

Entire page reformatted for readability and to add some things lost in the original Word conversion. Way overdue. Sorry.

INTRODUCTION:

This procedure explains how to hack apart a keyboard for use with MAME.

First, I have not completed this myself, but I have done all except the soldering so it should work.

Second, more than one person has fried their motherboard from a bad soldering job on a keyboard hack. If you are not comfortable, soldering on a printed circuit board, find someone who is. Use this procedure at your own risk and preferably test the controller on an older computer before plugging it into your brand new Mega-GHz PC.

Third, in deciding whether to use this method or a keyboard encoder, the most important considerations are in order: cost, available time, soldering skill, and number of **gameplay** buttons required. A typical keyboard hack will support two joysticks and between 8 and 14 total gameplay buttons. An almost unlimited (up to 104-keys) total inputs are supported, but these may have ghosting or blocking problems.

Finally, a keyboard hack is a very intensive solution for very limited results. Yes, it can be virtually free. Yes, it may provide enough inputs for most games in MAME for an arcade cab. Yes, MAME (and many other emulators) can get past the limitation of not being able to reprogram your controller to match it's default inputs. Having said that, I have come to the conclusion that a dedicated encoder such as the KeyWiz, I-PAC, or MK64 will be a better solution for the vast majority of arcade control users. At the risk of sounding like RandyT's (KeyWiz) posterboy, I don't think I could summarize my thoughts on this better than he did in a recent BYOAC post.

"I have a hard time recommending a keyboard hack to anyone who isn't chronically unemployed and to the point where every penny counts, or extremely bored and looking for a personal conquest. It's a lot of work, it's easy for a novice to ruin parts (like keyboard fuses on motherboards), and the end result is usually pretty sub-standard. The keyboard I hacked was noticeably poorer in performance than a dedicated encoder . . . I suggest you do a lot of reading before attempting this and don't base your decision on the successes or failures of a couple of individuals. This is going to require a certain amount of introspection on your part, after educating yourself, to decide if you are "up to the job". If you haven't soldered before, you'll need to practice. If you don't own a multi-meter, you should get one. It'll help tremendously.

For some, the task is trivial, but time consuming. For others, it is impossible and cost them a motherboard to find out. You should probably try to figure out which group you'll end up in before you get too far into it."

So now, if you're ready and still want to pursue this, let's get started.

GENERAL NOTES ON KEYBOARD CONSTRUCTION

The keyboard case is generally assembled in two halves. The halves are held together by (usually Phillips-head) screws. In some cases, plastic retainer clips are also used to hold the halves together. The keyboards that we are interested in will have a circuit card that the cable plugs into and one or more Mylar overlays that connect to this card and register keypresses. In some cases, a metal or plastic plate is screwed to the upper half of the keyboard to hold the Mylar against the keys.

The Mylar overlay will generally have carbon or copper contacts in a "double C" configuration like this:

One lead of this circuit (the left trace above) goes to a row of the matrix and the other lead goes to a column. The contacts may also be in the form of a split circle. On some keyboards, there are three sheets of Mylar. The upper sheet has conductive contacts on its lower surface and the lower sheet has contacts on its upper surface. A Mylar sheet without any contacts but with holes along the contact points is placed between the two sheets to ensure that only the selected keypresses register.

The Mylar overlay is generally clamped to a header terminal strip on the circuit card. On some keyboards, the Mylar overlay has terminal strips that correspond to the terminal strips on the keyboard circuit card and screws hold the Mylar to the card and ensure contact. This connection method would probably be easier to solder to.

WHAT TO EXPECT:

Using this procedure, you should end up with a controller capable of handling two joysticks and six buttons per player with no ghosting during game play. You may have to go through the entire procedure on a couple of different keyboards before you find a useable one, though. Once you find one that works, it should also be able to support 3-player, 3-button games, or 4-player, 2-button games as well.

TOOLS/MATERIALS REQUIRED:

1. Analog or digital multi-meter with resistance (ohms, W) scale ($10.99 at Pep Boys).
2. Soldering Iron, preferably low wattage.
3. Office style correction tape, single line.
4. Rosin-core solder, preferably the small-gauge electronics type.

5. Some type of wiring. I recommend IDE-type ribbon cable, available at Computer Stores (or surplus from old computer hard drive cables), Radio Shack, or Digi-Key (Digi-key P/N R007-5-ND Multicolor $5.09 for 5 feet, P/N R027-5-ND Gray $3.51 for 5 feet). I used 40-conductor cable because I will also be connecting my keyboard hack to removable control panels and to an optical controller board and 40-pin cable and 40-pin connectors are the easiest to find. If you are just connecting the keyboard hack to a single arcade controller, you could probably go with 24-conductor cable, which is a few pennies cheaper and will make a cleaner installation. You also can peel off the extra conductors if they are not required.

   BTW, I wanted solid rather than stranded conductor wire for my project; because I thought it would be easier to work with. Digi-Key does not carry this. They gave me the 800 number to 3M. 3M only sells it in 200-foot rolls for a lot of money. However, they said the local distributor might be able to get me a sample of it to try. I called the distributor and he sent me a five-foot sample, Priority Mail, for FREE. I don't know if this will work for you, but it's worth a shot.

6. Cheap Keyboard - Only the Integrated Circuit card from the keyboard will actually be used, so don't spend a lot of money on this item. The choice of a DIN-5 or PS/2 keyboard is not important, as there are adapters to go either way. However, do NOT buy a USB keyboard (and do not hook your PS/2 Keyboard up to the USB port) as the standard USB interface only supports six simultaneous keypresses (plus any modifier keys such as Alt, Ctrl, Caps Lock, or Shift). Also, the six-key limit is not merely a keyboard manufacturer's decision but a function of the Windows keyboard driver. Any attempt to press more than six keys simultaneously will result in the infamous "Blue Screen of Death" and require a re-boot. (This is not a problem with the I-PAC in USB mode, because the inventor wrote special code to get around this limitation.) The following keyboards have been evaluated (in alphabetical order): (If you use a different keyboard, e-mail me the particulars, and I will add it to this document.

   A. Apricot Computers Ltd.,Mitsubishi Electric UK, FCC ID : AQ6-MTN4XZ15, Model : RT6656TUK, P/N : 121287-002 REV.D, S/N : 31260302. Thanks to Oliver Wells for the data on this keyboard. The matrix is included as Appendix C.

   B. Compaq model: SK-2800c, Compaq p/n: 341162-006, Spare p/n: 387084-003 Thanks to Mr. Arcade for the information on this keyboard. The matrix is included as Appendix D.

   C. Digital Research Model DR-104 KEY (box art), Model Scorpius 104 (keyboard itself), Circuit Card - Qtronix Qwin95-A.PCB 27-050-150, Microcontroller - UMC UM6868A-0011 9903M EBNC0. This is a DIN 5 104-key keyboard with included PS/2 adapter. Screws hold the halves together and a plastic plate holds the Mylar to the upper half of the keyboard. The circuit card is approximately 4.5 by 2.5 inches and the Mylar sheet is clamped to it. It uses a 16x8 matrix and key blocking. The matrix is included in Appendix A. I purchased it for $11.99 with an $11.00 rebate from Best Buy in December 1999. Digital Research is bad about honoring their rebates, though.

   D. Micro Innovations - model KB400i. Thanks again to Mr. Arcade. The matrix is included as appendix E.

   E. Mitsumi KPQEA42A - This is a DIN 5 104-key keyboard. Screw and plastic clips hold the halves together and a metal plate covers the Mylar sheet. The Mylar sheet appears to have three wires which each key press contacts. I have not mapped out the matrix, but the three wires may make this keyboard unusable, depending on function. The circuit card is approximately 1.25 by 3.25 inches and the Mylar sheet is held against it with two screws. It uses key blocking. I purchased it with a computer system from a local computer store in June 1998.

   F. Non-Branded Keyboard, chicony ver a 105-06868-010 6868A-0017 9947M FADD3 keyboard encoder. Thanks Pete Clements for the info. Matrix included as Appendix F.

   G. PC Concepts PC104 Keyboard Model Number KWD-203, Item Number 61595, Circuit Card - P/N 001-00200-001, Microcontroller - Chicony C34451BE Ver L 105-08049-250 9807 CT1. This is a DIN 5 104-key keyboard with included PS/2 adapter. Screws hold the halves together. This keyboard uses the two Mylar sheets with a Mylar separator. The circuit card is approximately 4.5 by 2.5 inches. The column inputs are on one edge of the card and the row inputs are on the other edge. It uses a 16x9 matrix which casts serious doubts on my later comments about a 24-terminal matrix being "almost an industry standard" and it uses key blocking. The matrix is included in Appendix B. I purchased it at Office Depot in November 1998 for $14.99 with a $15.00 rebate.

   H. Silitek SK-1500. This is a 107-key PS/2 only keyboard. Screws hold the halves together. The circuit card is approximately 1.25 by 3.25 inches and the Mylar sheet is held against it with two screws. I have not mapped out the matrix yet, but this keyboard looks like it would be very easy to hack. It uses key blocking. I purchased it with a computer system from a local computer store in May 2000.

7. Terminal Blocks or Headers - Optional but highly recommended. For removable control panels, such as mine I plan to use a Digi-Key MPL40K-ND Plug Connector ($8.05). This attaches to the IDE-ribbon cable without soldering and allows quick connection/disconnection of panels. For permanently installed panels, I recommend small PCB terminal blocks Digi-key P/N ED2236-ND ($2.07 for 12-pin). I like these because they are compact and reasonably priced. The twelve-pin TB is 2.25 inches long by 9 mm wide by 12 mm high. Using terminal blocks will make it much easier if you decide to change button layouts later, or for example, if MAME changes coin input keys from 3 to 5, as it did a few revisions ago.

8. Secondary keyboard - If the keyboard to be hacked is your primary keyboard, a second keyboard or a working different computer system is required, to allow the use of Word or EXCEL, etc.

9. Keyboard Testing Software - (Freeware) - This allows you to see what keys are really going to the computer when you press the keyboard keys. Go to Saint's BYOAC site and follow the links to downloads and utilities. The advantages of each program will be explained in the testing section of these instructions.

ALTERNATIVES (AND THEIR PROS AND CONS):

Okay, if you read through this procedure and have decided that hacking a keyboard is not for you there are two other options: a gamepad hack, or a pre-built encoder. Detailed information on either option is available here. Also, I will include some brief comments below.

1. Gamepad hack - Physically a gamepad hack requires about the same amount of work as a keyboard hack, i.e., you still have to tear the pad apart and solder wires to the circuit contacts. You don't not have to map out the matrix and you don't have to worry about ghosting. Another advantage is that if you hack a Playstation gamepad, you can use it with adapters with any type of console or PC games. The biggest drawback to a gamepad hack is that you are limited to a usual maximum of 14 inputs. You could hack multiple USB gamepads to get more inputs, but many people have reported problems with them "swapping" positions when the computer is powered off and back on.

2. Dedicated Encoder - The reference page provides much more detail, but the main three encoders I will recommend (alphabetically) are the I-PAC, KeyWiz, and MK64. I will not recommend one of these over another, because they all have different features and it depends on the combination of price and features that are important to you. However, all of them feature common-ground rather than matrix-mode connections, more inputs than a keyboard hack, no possibility of ghosting, better response time, and full programmability.

KEYBOARD HACK ADVANTAGES:

1. Cost - Forget about wire and terminal blocks because you would have to buy these either way (Well okay, no TB's if you use a dedicated encoder and don't have removable control panels). If you have basic electrical tools, your total cost is the keyboard, which you can often find for less than $5.00 with a rebate or for free (if the keys, case, etc. is broken) at computer surplus stores. In a worst case, at $15.00 for a keyboard, $10.00 for a multimeter, and $10.00 for a soldering iron, the total cost of $35.00 is still less than any other option, and the cost of doing a second one is only $5 or free. However, if you are only building one project, haven't soldered, and don't own a multimeter, you are probably much better off buying a dedicated encoder.

2. Number of total inputs - As previously stated, a keyboard hack will allow a total of 101 (or 104) separate inputs, however only probably 16 of these can be pressed simultaneously. This is more total inputs than any other option (and more than you would ever need).

3. Cost - This is the cheapest way possible to connect arcade controls to your PC.

4. Feeling of Accomplishment - It's pretty fun knowing that you created something yourself and it does what it was intended to and works better than anyone says it should. Especially when most everyone you ask for advice says "Save yourself the headaches. Buy a dedicated encoder."

5. Cost - Did I mention it is really cheap? If not, see items 1 and 3 above.

6. Size - In some cases, the circuit cards on newer keyboards measure approximately 1.25 by 3 inches. This is about half the size of most dedicated encoders and is smaller than any of them. This may be an important consideration for a desktop controller with many buttons.

KEYBOARD HACK DISADVANTAGES (REAL AND PERCEIVED):

1. Soldering - Be sure to read the warning at the beginning of this page about frying your motherboard. Actually, all soldering is done to the input side of the integrated circuit, so I think you would be more likely to end up with stuck keys or non-working keys from a bad soldering job. Of course, if the solder drips onto the IC circuit board and shorts out a couple of pins. . . Well let's just say that could be bad. Basically, if you don't know how to solder well, and I don't, you are probably better off letting someone else solder this, as I plan to do.

   Tip: if a keyboard hack goes wrong, i.e., the 5 volt supply is grounded, the motherboard keyboard port will appear to have died instantly. This is because most motherboards supply the 5 volts through a tiny fuse which blows rather easily. While expensive to professionally replace, the fuse can pretty safely be wired across, saving an apparently useless board. Keep in mind that this now leaves the other areas of the motherboard vulnerable, so if the original problem is not corrected or another faulty keyboard hack is connected, more extensive damage may occur. Also, many A-Open brand motherboards use a resettable circuit breaker in lieu of a fuse on the keyboard and USB ports. While A-Open typically offers a good selection of features per dollar and their boards have been well-rated in most reviews, I would not recommend buying a motherboard solely because of this feature. However, if you like their products, this is something to consider and maybe something to look for in other manufacturer's boards.

2. Time - For me, I would rather spend time on a project than money. Those that disagree, should buy one of the alternative products shown above. I would estimate the time required for this project as follows: disassembly - 0.2 hours, mapping out the matrix - 2.0 hours, reassembly - 0.1 hours, planning the input keys - 1.0 hours, testing the proposed input keys - 0.5 hours, second disassembly - 0.1 hours, soldering (rough guess, from here on I have not accomplished this) 1.5 hours, mounting - 0.5 hours, miscellaneous and sundry distractions - 1.1 hours. Total time: 7.0 hours (if my calculator is correct). If you happen to find or purchase one of the keyboards mapped here, you can shave most of the key mapping and planning stages, so the total time should be about 3.0 hours.

3. Number of Independent Inputs – As explained elsewhere a keyboard hack will support from 12 to 18 independent inputs. This is sufficient for two joysticks and between eight and 14 total buttons. The Player Start, Coin Input, and other maintenance keys are not included and may in fact have blocking problems. Normally the likelihood of this happening is rare as most games do not allow the 1P or 2P Start keys to be depressed during gameplay. (It can be made even more rare by assigning the Coin 1, 1P Start, and 2P start keys to keys not in the same matrix column as any of the joystick direction keys or the B1 or B2 buttons for each player, but this added complexity is probably not required, unless you cannot **EVER** tolerate ghosting/blocking problems.) If you require additional buttons for gameplay (highly unlikely) or if you cannot tolerate even the **possibility** of ghosting with the coin or start keys, you should not use a keyboard hack.

   Also, Andy Warne (I-PAC) pointed out and Randy T (KeyWiz) confirmed that most current keyboards employ two blocking methods. There is the key blocking routine mentioned below, but there is usually a pressed key maximum hard-coded as well which varies but is usually eight. This is done to reduce the amount of RAM required in the keyboard microcontroller. I have no doubts that this exists, but both the keyboards that I have mapped have been able to register at least 15 simultaneous keypresses (I ran out of fingers on the PC Concepts verification). Unfortunately, there is no easy way to tell and no easy solution if your keyboard has this limitation. You basically have to map the matrix, test the keys, and if keys stop registering either limit yourself to that many keys, or start over with a different keyboard.

4. Ghosting, Masking, and Key Blocking - This is probably the single, strongest reason why people avoid the keyboard hack. There is also a lot of outdated or plain wrong information about this on the web. It is so important that I will spend most of the next section (Common Misconceptions) covering the technical aspects of it in detail. A short summary of the problem and solutions is included here:

   A. What happens - In a nutshell, newer keyboards work by scanning a matrix. See Appendix A for example keyboard matrices. When three keys that form the corners of any rectangle anywhere on the matrix are depressed, one of two things happen, either the keyboard will assume that the key marking the fourth corner of the rectangle is also depressed and send this key to the computer (key ghosting), or the third key will not register (key blocking). In addition, if ghosting occurs, if the fourth key of the rectangle is depressed simultaneously and one of the other keys is released, the key release will not register (key masking).

B. How serious is this, really - That is a tough question. Keep in mind that this condition only occurs when three or more keys are pressed **simultaneously**. I played Asteroids for about six months before I read a review of the Stick-It on Saint's BYOAC site saying that Windows popped up when he pressed alt (thrust) and space (hyperspace) at the same time. I tried it at home and it instantly did the same thing, although I had never noticed it before. Also, consider that a two-joystick game with both joysticks on the diagonals generates four simultaneous keypresses without any other buttons being depressed. The trick is to pick your keys carefully and avoid the problem altogether. BTW, the asteroids problem was a windows shortcut problem and not key masking, and was easily solved.

C. Why is there so much confusion: Basically, and I am guessing here, about 1989, keyboard chip manufacturers became aware of the ghosting problem, decided that having uncommanded keys show up was a bad idea for word processing and went to the blocking route. At about the same time, MAME took off and people using older keyboards discovered that having uncommanded keys show up for emulators was a bad idea. They recommended the use of diodes to solve the problem, which worked for some of the key ghosting keyboards. Other older key ghosting keyboards used very low voltages for the switch scanning and thus were unaffected by diodes. Later, people bought newer keyboards and found that diodes didn't make any difference and they were still having (now blocking) problems. A huge debate ensued and is still raging about whether diodes make a difference, and when or if they should be used. Hopefully, I will not respark that here.

D. What are the solutions: Ironically, the solutions end up being identical, at least as far as I am concerned. If you have one of the older keyboards that actually demonstrates ghosting instead of key-mapping (and I don't think you will find a new keyboard which does), you could (theoretically) attach diodes to each input line and be able to use all 101-keys without ghosting (assuming the keyboard does not limit simultaneous inputs in firmware). However, this doubles the amount of soldering required, and increases the possibilities of a solder connection coming loose and messing up your computer. Additionally, if this works, I would expect Hagstrom to be aware of this solution and incorporate diodes in their encoders, and the fact that many people are posting suggested key assignments to use with Hagstrom encoders in matrix mode tells me that they still have ghosting/masking. If your keyboard uses key blocking, diodes will not help and may actually hurt things. However, in either case, if you choose your keys strategically, you will not have any problems with ghosting or masking, regardless of keyboard type.

5. Lack of a keyboard when using the controller (no keyboard pass-thru support). Some people mention this as a problem with keyboard hacks. Actually there are several solutions. Before I get to them, consider the following:

A. If you are using your controller in a MAME arcade cabinet, you shouldn't need a keyboard at all except for set-up and configuration changes, or for adding new games or new MAME releases. By designing the cabinet for this, it should be fairly easy to shut down the computer, unplug the keyboard hack, plug in the standard keyboard, make the updates, and plug the keyboard hack back in. Or you could use a wireless keyboard for maintenance.

B. Now, in order of my personal preference, here are the solutions to the no-real keyboard problem:

1. USB - if your computer supports USB, purchase a USB keyboard for general use and plug the keyboard hack into the regular keyboard port. You should be able to find a USB keyboard for around $15.00 or less. Under NO circumstances should you use a standard keyboard for general use and hack a USB keyboard for MAME, because the USB specification does not allow more than six simultaneous key presses (plus modifier keys). This does not apply to the I-PAC in USB mode. (Thanks to Andy Warne for this tip.)

2. Keyboard splitter circuit: Saint's BYOAC site has links to two keyboard splitter circuits: Stephan Hans, and one from everyday practical electronics. Either one could be built for under $10.00. The following information regarding these circuits was provided by Bill Lash in response to questions I posted on the BYOAC message board and should be helpful for someone considering these options:

a. What are the differences - Stephan's circuit is set-up so that one keyboard is primary and the other is secondary. The LED's on the primary circuit will always work and on the secondary one they never will. If you are building a MAME Cabinet with LED's on the panel, plug your keyboard hack into the primary port and the computer keyboard into the secondary. If you are building the controller for a general use computer that also plays MAME, plug you computer keyboard into the primary port and your keyboard hack into the secondary one. The other circuit is easier to build and costs a few dollars less to build. It will activate the LED's on the keyboard or hack which last sent an input to the computer. Thus, your LED's in MAME will not work until you press a button and your LED's on the computer keyboard will then be off until you press a key on it.

b. In both circuits, the keyboard port pins are numbered looking at the cable side in on the receptacle (looking at it the way the cable would be inserted). Also notice that the simpler circuit shows a plug rather than a receptacle for the computer connection.

c. Integrated circuits (IC's) are numbered as follows: Pin 1 is indicated by a dimple or U-shaped indentation on the side of the chip that contains Pin 1 (or by markings on the IC). Looking down on the chip, Pin 1 would be on the upper left corner and pin numbering continues CCW down the edges and around the chip, so that the highest pin number ends up at the upper right corner opposite Pin 1.

d. The simpler circuit shows a 4093 IC in two different places on the circuit schematic. Actually, only one 4093 IC is required.

e. The simpler circuit implies that +5V and ground are coming in on a external wire into the circuit. In fact, these inputs are supplied through the keyboard port.

3. Keyboard switch: Lew's wheels includes a circuit using a SPDT (single pole dual throw) switch and three sockets to switch between two keyboards.

4. The simplest solution is to hot-swap between your computer keyboard and your keyboard hack. I asked Andy Warne about this and he said that theoretically there is a danger in this, but he had done it at least a hundred times setting up the I-PAC and never had any problems. I also had tried it several times with no problems before I asked him. Digi-Key (among others) sells an M-F keyboard extension cable which will allow you to do this without crawling around behind the computer.

5. Y-Key-Key: Also on Saint's BYOAC site, PI Engineering sells this adapter for about $50.00 which will allow you to have two keyboards connected to your computer. However, as far as I can tell, the device operates identically to Stephan Hans circuit and the price will eat up any of your savings by going the keyboard hack route.

6. Micro-center (among others) sells a cable splitter (for $7.95) for laptop computers which allows a keyboard and mouse to be hooked up to the same port. The cable has three connectors, a female PS/2 and two male PS/2 's which are labeled keyboard and mouse. There were no instructions as to whether it was intended to plug into the PC keyboard port or the PC mouse port. I tried it in various configurations with two keyboards, couldn't get it to work, and returned it. There might be a way to get this to work, though. If someone finds one, let me know and I will include the information here. This is a similar device which is designed for connecting two keyboards, but I have not tried it and don't know if it works or not.

6. Wiring complexity - Dedicated Keyboard encoders in non-matrix mode and also the standard arcade machine JAMMA harness reference all keys to one common input. Thus, you can string one wire to the second terminal of all of your arcade controls. In matrix mode or with a keyboard hack, each button connects two of the input contacts to the circuit card. This really doesn't make any difference, but it does require a different way of thinking about wiring.

7. Lack of programmability - A keyboard hack is non-programmable. On the DR-104 keyboard (for example), there are only 16 keys that can be pressed simultaneously. And only certain keys may be selected for those sixteen keys. There is definitely, however, more than one choice of key assignments for any particular key. For example, looking at Appendix A, in Row 12, I have assigned the Space bar to P1 B3. I could just as easily have assigned this to another button, say P1 B2 and changed the P1 B3 assignment to a key in a different row. I also could leave all the other keys the same and assign Keypad (KP) 7, KP4, KP1, Num Lock, Down Arrow, or Delete to P1 B3 instead of Space and the hack would work just as well. However, I cannot assign space to one button say P1 B3 and another Row 12 key, say Down arrow to P1 Down without the possibility of ghosting/blocking. Different keyboards will have different acceptable key assignments, but the concept is the same. Since MAME allows reassignment of all input keys, this is not a problem (except for some awkwardness in the game setup) but with other emulators, it might be critical.

8. Available key assignments - Similar to the above problem, you will have some problems with key assignments in MAME. I would be highly surprised, and it would be only blind coincidence, if you found any keyboard hack which will support two joysticks and 12 game buttons and map the joystick to the direction arrows. On the other hand, of the default Player 1 Buttons 1 through 6, I would expect at least four of the six to be available, regardless of the type of keyboard. The DR-104 key (for example) supports all six buttons.

9. Build Quality - I mean here the quality compared to a dedicated encoder. Consider the following two items:

   A. A dedicated encoder is mounted to a PCB with a terminal strip that the input wires attach to, and a detachable cable to connect to the PC. There is very little possibility of damage. A keyboard hack generally has the keyboard cable soldered to the PCB with a mess of wires soldered to the inputs sides. The entire assembly is much more flimsy. Granted this shouldn't matter in an arcade cab where the hack will not be moved much, especially if you use terminal blocks after the input cables, but it's still a consideration.

   B. Let's say either option "dies" for whatever reason. With a dedicated encoder, you just purchase a new one, unwire the old one, and wire the new one up to the same wires and you're finished. Even if the original dedicated encoder is no longer available, (assuming you recorded your settings), you could buy a different model, wire it up, and re-program it to match you old settings in under 45 minutes. If your hack dies, you probably won't find the exact same model keyboard, so you have to start over at square one - disassemble, re-map the matrix, re-test, assign different inputs keys, solder new wires, re-connect new wires, and re-configure MAME to match the new inputs.

10. Lack of shift key support - The I-PAC, KeyWiz, and MK64 encoders include several special "shift key" functions which are designed to reduce the number of buttons required on the control panel. For example, on the I-PAC, pressing the 1 key and the 2 key together simulates ESC and exits MAME. Since R36B13, MAME allows key combinations and multiple assignments to be made to each input. Therefore, from the MAME Input (General) menu, selecting UI CANCEL = "1 2 or ESC" accomplishes the same thing. Caveat One: The dedicated encoders include this functionality in the controller code, so this feature will also work using it on emulators which do not support key remapping. Caveat Two: Key remapping in MAME does not exactly mimic the I-PAC and MK64 shift functions. The differences and the results are as follows:

    On the I-PAC or MK64, if you press the shift key nothing happens until you release it. Then it either sends its own code (1) or (if you have pressed another key) the shift code. If you map the same combinations in MAME, you will always get the first key send followed by the combination key send. The effect of this for each shift key function is as follows:

    A. Coin Input: The I-PAC uses 1 and P1B1 keys to mimic coin input. In MAME, from the attract screen, if you map this combination, the first coin input will register. If you try to input a second coin, the game will start a 1-player game, because MAME reads the 1 key first. Therefore, you will never be able to start a two player game. For this reason, I recommend having a dedicated coin input key on your panel if using a keyboard hack. (You CAN get around this from the attract screen for most games by pressing the P1B1 key FIRST, followed by the 1 key; but a dedicated key is the easier solution).

    B. ESC: The I-PAC uses the 1 and 2 player start keys to mimic ESC. In MAME, the game will either quit immediately or will start a one or two player game and then immediately quit. This is perfectly acceptable and I recommend mapping these keys if you don't want a dedicated escape key. However, a drawback to both of these methods is that you might exit your game if both players press the start buttons at the same time.

    C. Pause: The I-PAC uses the 1 and J1 Down key to mimic Pause. In MAME, if you are on the attract screen with credits already input, MAME will start a new game before pausing the game (assuming you press the 1 key first). If you are in one of the game playing screens MAME will pause properly unless you press the J1 Down key first, in which case the player position will move down slightly before the game pauses. This is acceptable, although I personally prefer a dedicated Pause key so that I don't have to remember key combinations when I need to stop gameplay at a critical point.

    D. Tilde and Tab: The I-PAC uses the 1 and J1 Up or J1 Right keys, respectively, to map these functions. These keys will function the same way as the Pause key above. See the section on assigning keys for my personal key mapping recommendations on these keys.

    E. Enter: The I-PAC uses 1 and J1 LEFT to mimic this key. This will work in MAME. This key is only used in MAME for items on the Configuration or On Screen Display Menus. The only difference between using this key in MAME is that if you enter these menus without pausing the game, the player may move left while you are hitting enter! You are not likely to do this, though. Again, see the key mapping section for my personal recommendations on this key.

LITTLE KNOWN FACTS AND COMMON MISCONCEPTIONS:

1. Matrix Mode:

   Most keyboards use some kind of matrix to interface with the keyboard IC. This keeps the manufacturer from having to run 105 inputs (104 keys + common) to the controller. Instead, only 25-30 inputs are used, by assigning the inputs to rows and columns and assigning keys to each combination on the matrix, all of the keys are supported. For some reason, 24 input matrices are almost the industry standard. For example, the DR-104 keyboard has 26 terminals connected to the IC. However, terminals 17 and 18 are used to support the Windows 95 menu keys only. Since these keys are not recognized by early versions of MAME and most all other programs, this results in an effective matrix of 16 x 8 = (you guessed it) 24 keys.

   Ideally, one would like to find a matrix with 22 x 2 (rows by columns). However, this is impossible since such a matrix would only support 44 total keys. The best keyboard we could hope to find would be an 18 x 6 matrix (108 keys theoretically supported). This keyboard would support two joysticks and 7 buttons per player (or two joysticks, six buttons per player and the player 1 and 2 start keys), with no chance of ghosting. If someone finds a keyboard with this configuration, let me know and I will add the information to this document. The worst option would be a 12 x 12 matrix. This will only support two joysticks and 4 buttons per player (or one joystick, six buttons, the 1P, Coin Input, Pause, and ESC keys) with no chance of ghosting. As you can see, even this is not a bad option, unless you really like the Mortal Kombat / Street Fighter style of games.

2. Ghosting, Masking, and Blocking:

   Dave Dribin does a much better job of explaining the technical side of key ghosting and masking than I could ever hope to. Stephan Hans also has some excellent info on his site. Follow the links to SuperDuperArcadeConsole (using Cherry Core). Stephan's page doesn't provide much detail on how you pick the keys to avoid ghosting (that's my job), but he uses the same concept that I am and his site was the first place I ever saw this mentioned. I think his information about diodes is incorrect, though :-((. Thanks to Saint for the links. I will try to cover the **practical** side of the problem here:

   A. Key Ghosting - Most keyboards use a matrix to determine key presses. The DR-104 uses a 16 by 8 matrix. To show a random real example configuration:

	19	23
1	E	C
3	Q	X

   Ghosting may occur when you press any three keys that form a rectangle (**any number of rows and columns apart**) in the matrix. In the example above, pressing E, Q, and C, simultaneously will generate X. This is rarely a problem as the X key, for example is not usually used by another game input and all keys need to be pressed simultaneously.

   B. Key masking - Since the keyboard doesn't know whether C or X should be pressed and generates both keystrokes, it also doesn't know when the key is released, and will keep registering the keystroke even after the key is released. This is called key masking and will result in an apparent stuck key during gameplay, which is a problem.

   C. Key Blocking - On the DR-104 keyboard, and I believe virtually all newer keyboards (from about 1993-up), a technique called key blocking is used. In this method, the IC scans the matrix and blocks any keys which could create ghosting or masking. Thus when any three keys which would form a square if a fourth key is pressed are depressed, the third key will not register. Using the example above, when E, and Q are both depressed and C is then pressed, the C key will not register.

      Also, any keys pressed after the three conflicting keys are pressed will not register until one of the three keys are released, releasing the blocking condition.

      The following message from the BYOAC message board, from Bill Lash to Malberg, documents this and is preserved here for historical purposes:

<Malberg wrote> I noticed while I was laying out my matrix for my keyboard hack that it was a 16x8 matrix. Just for the sake of testing the matrix for ghosting, with my second identical keyboard from Compaq, I picked any 3 points on the matrix that form a rectangle or square and pressed them. I was using the Ghostkey program which I downloaded from this site to test for ghosting. As I was pressing the keys one at a time and holding the previous key down, I was only able to get 2 keys of the rectangle to only show. And I tried many combinations and different rectangles, anything I tried I only got 2 keystokes to show. But If I pressed down any keys that didn't form a rectangle then I was able to hold down 8 to 10 keys and all would show in the Ghostkey program.

Can anybody give me any suggestions why this is so???

I thought that if you hold down 3 keys in a rectangle the forth key would automatically be triggered which is called ghosting!!! I can't give get 3 to show in a rectangle, can someone please explain!!!

: Regards -

<Bill Lash replied>This sounds like they are using a software workaround for the ghosting problem. The keyboard chip is basically a little processor that scans the matrix and sends the data to the PC. Someone got clever several years ago, and decided that you don't need diodes in the matrix (the classic way of solving the n-key rollover or "ghosting" problem), if you detect that 2 keys of the rectangle have been pressed, you can have the software ignore any new corners of the rectangle that get connected until you drop back to only one or zero corners being detected. I see this method mentioned in an application note from National Semiconductor from 1989 for using one of their COP microcontrollers as a keyboard encoder. It seems that a lot of newer keyboards use this method, which makes diodes useless. I think there are some older keyboards that don't do this, and they are probably better to use for a hack.

You may have found a good way to determine if a keyboard encoder could benefit from diodes. If you actually see ghosting, and not the 2 key only behavior you see now, the ghosting can be taken care of with proper use of diodes. If you see the 2 key behavior, diodes won't help. You may be able to come up with a clever map for your controls to get enough free inputs though. Good luck.

BTW, the entire national semiconductor paper referenced above can be found by searching the archives at BYOAC message board for "national semiconductor" and opening this thread by Bill Lash entitled "Re: Diod Method A Myth?"

3. Diodes

   There has been much heated debate on the use of diodes in a keyboard hack. It has been very ugly at times. I don't want to rehash that here. Diodes may or may not work on older keyboards which show key ghosting. They won't work on keyboards that use key blocking. They are not required on either if you pick your input keys carefully. See the FAQ section for more info on diodes.

HOW GOOD ARE THE RESULTS:

Since the DR-104 keyboard uses an 18x8 matrix, I should theoretically have eighteen keys available with no blocking. In fact, this and many other keyboards put the right and left Windows Menu keys all alone on two rows and since earlier versions of MAME do not recognize these keys, only 16 keys are available. However, since a joystick cannot be both up and down or both right and left at the same time, these keys can be allowed to block and only two inputs per joystick are required. Thus, my method allows two joysticks and six buttons per player to operate with no chance of key blocking. I still will get blocking if I press the 1P, 2P, coin, or pause keys with certain other keys depressed, but this should rarely create any problems. I was even able to use the MAME default keys for the Player 1 inputs, although the joystick and Player 2 inputs are nothing like the MAME defaults. Later, I was able to expand this concept, and the keyboard now also supports 3-player 3-button games and 4-player 2-button games.

HOW TO DO IT:

SECTION A - USING ANY TYPE OF KEYBOARD.

If you have one of the example keyboards for which the matrix has already been mapped, you can skip most of these steps. See SECTION B for the procedures to follow: During these procedures, I will use the DR-104 keyboard in developing the examples, however, the general procedures should apply to any keyboard.

1. **Tearing it down:**

   A. Place the unplugged keyboard to be hacked face down on the desktop and remove the screws holding the keyboard halves together. (Generally, eight to ten philips-head screws. Some of the screws may be under labels or some keyboards use plastic retaining clips to hold the halves together. If the halves don't separate easily, look for this). Do not lose the screws as you will need them again later.

   B. Carefully, remove the back cover from the keyboard. Depending on the design, this may leave all of the little key buttons and rubber pads which go under them loose. Do not lose these either. You will see one of the following three things:

1. There may be a bunch of metal plates inside the keyboard. On older keyboards these were used for magnetic shielding. I have never run across one of these, but one of the example websites said that they cannot be hacked. However, the Mitsumi keyboard uses a metal plate to cover the Mylar membrane, so you might try removing the metal plates and seeing if you see a Mylar sheet. Otherwise, trash the keyboard, buy another one, and start over.

2. The keyboard may have a huge circuit **board** which covers all of the keys and little mechanical switches below each key. These were typically used on the old IBM-type "clickety-click" keyboards. Many people have reported that these keyboards did not use a matrix and all keys could be pressed simultaneously. Andy Warne reported that these keyboards do use a matrix and therefore, I would expect some type of ghosting/masking behavior. All I can recommend is that before you start, you check for the presence of ghosting/blocking using one of the keyscanning programs as shown in Step 5. If ghosting/blocking is not present, you connect the keyboard by just soldering two wires to the back of the circuit board opposite the mechanical switches for the keys you want your buttons to use and running these wires to the appropriate switches. In some cases, people have reassembled the keyboard after they finished and still been able to use it as a regular keyboard and as an arcade control!

3. The keyboard may have a circuit card in one of the upper corners (follow the input cable) which connects to a Mylar sheet(s) which records the key presses. The Mylar sheet may be retained under a plastic or metal cover plate. Remove the plate so that the Mylar sheet and circuit board are exposed.

2. **Mapping the Matrix:**

An alternative method which many people recommend involves disassembling the keyboard and plugging the circuit card into the keyboard port and shorting between the terminals and recording the generated keystrokes to map the matrix. I recommend my method over this one for the following reasons: 1) In my method, the keyboard is unplugged so there is no possibility of damage to the computer system during testing. 2) It's not always easy to tell where the connection points are and which ones are rows and columns. I'm not sure what happens if you connect two row inputs together, but it might not be good. 3) Many inputs (L Alt, Windows Key, etc.) won't display on a screen like NotePad. (I realize GhostKey or some of the other utilities will work; however, even some of the utilities can't pick up the NumPad keys, or the difference between R Alt and L Alt. And if it doesn't detect these keys, they will be left out of the final matrix. With my method, you are using the physical locations on the keyboard, so every key gets picked up, and no effort is wasted). 4) A 16x8 matrix is 128 inputs, but MAME can only use 104, so you have 24 unused and unusable inputs that you still need to map out. (Add another 36 to that for an 18x8 matrix). 5) If the keyboard ends up being a poor choice to hack, with my method you can reassemble it and either use it as a regular keyboard or yard sale it. With the other method, you have spent a couple of hours soldering wires to a keyboard encoder that you will never use.

I recommend using either a spreadsheet program or a word processing program to map out the matrix, but it can be done on graph paper as well.

A. Pick one terminal of the circuit card to be terminal one. Label the first line of your spreadsheet or word processing file as "Terminal one is right edge of card (connector side of board) looking at component side of board." or similar wording. You may also use correction tape to label terminal 1 on the Mylar sheet so that you don't get confused as you start mapping out the matrix. This ensures that you will still know how to wire everything up when your joysticks and buttons arrive three weeks later.

B. Select the top row and the first column of your table or spreadsheet to **bold.** This will make it easy to distinguish between row 5 and an input for key 5 on the keyboard.

C. On many keyboards, the Mylar sheet(s) will have more contacts than the actual number of keyswitches on the keyboard. This usually occurs around the shift or bottom row keys, or around the enter key. I think it occurs so that the keyboard manufacturer may use the same Mylar with L-shaped or straight Enter keys, etc. These contacts are not required for the keyboard being tested and might be confusing. Cover the contacts with correction tape on the conductive side of the Mylar to ensure they are not mapped.

D. Lay the Mylar sheet (and possibly the circuit card also) flat on the desktop as follows:

1. If the Mylar sheet(s) and circuit card are connected (clamped) together, remove them as an assembly and lay them flat on the desktop with the conductive side up. (The same orientation they would normally be in when you actually use the keyboard.)

2. If the Mylar sheet is held down by the screws holding the circuit card to the keyboard, remove and retain the screws and place only the Mylar sheet on the desktop with the conductive side up. (The same orientation they would normally be in when you actually use the keyboard.) In this configuration, (looking at the disassembled upside-down keyboard), the circuit card is usually mounted with the component side and the contacts pointing up and the Mylar sheet is placed on top of the terminals with its contacts face down. Thus if you label the circuit card terminals from 1-26 starting at the left, you must label the Mylar sheet from 1-26 starting at the RIGHT when the sheet is flipped over.

E. You will now have a Mylar sheet with contact points corresponding to the key positions on the keyboard under test. These should be similar to your standard keyboard so you may use it to determine the key associated with each contact point. The contacts will look similar to the following

"double-C" configuration, or possibly a split circle configuration. You will want to check for continuity between each leg of the C and the traces leading to the circuit card as follows:

1. Turn the Ohmmeter (multimeter) to the highest resistance scale.

2. Touch the positive and negative ohmmeter leads together. You should see full scale deflection of the needle reading. (A digital Ohmmeter should read 0.0).

3. Touch one lead to each side of one leg of the double-C trace as shown by the red dots below. You should see full scale deflection. This will ensure that you are testing on the conductive side of the Mylar.

4. Keep one lead on one leg of the Mylar contact and touch the other lead to the traces leading to the circuit card, in order, until you see full-scale deflection. Repeat for the other leg of the contact. For example (on the DR-104 keyboard), for L CTRL, you would see full scale deflection on pins 4 and 25.

F. Test all keys and complete the matrix as follows: Place the lower number terminal on the row and the higher on the column. Insert rows and columns in order as necessary. (I usually just work left to right and up the keyboard, but it doesn't really matter. These examples are from the DR-104 Key keyboard).

1. For the Left Ctrl key (contacts 4 and 25):

	25
4	L Ctrl

2. For the L Win95 key (contacts 17 and 20):

	20	25
4		L Ctrl
17	L Win	

3. For the L Alt Key (contacts 10 and 22):

	20	22	25
4			L Ctrl
10		L Alt	
17	L Win		

4. For the Space Bar (contacts 12 and 22):

	20	22	25
4			L Ctrl
10		L Alt	
12		Space	
17	L Win		

5. For the R Alt Key (contacts 10 and 24):

	20	22	24	25
4				L Ctrl
10		L Alt	R Alt	
12		Space		
17	L Win			

6. And so forth. . . Continue this process until all of the keys are mapped. When complete, add rows or columns for any unused circuit traces. Usually there won't be any.

G. You should now have a table similar to the matrix in Appendix A. The only other possibility is that if you chose the wrong end of the card for terminal 1, your table may have more columns than rows. In that case, you can either use the transpose function of your spreadsheet or word processor (save and print the current configuration first), or you can leave it as it is and just substitute column for row in the rest of this document.

3. **Sizing things up:**

A. At this point, you can determine how successful a keyboard hack will be for you. Look at the Matrix and eliminate any rows which only contain keys not used by MAME (the Win95 and Win98 Powerkeys (later versions of MAME support these, but I don't recommend using them)). Count the remaining rows. The total will between 12 and 18 rows. In our example, the result is 16 rows.

B. The total number of rows (16) corresponds to the maximum number of non-ghosting keypresses which the keyboard controller allows. Since a joystick cannot be both up and down or left and right at the same time, these keys may be placed on the same row and allowed to block each other at the same time. Thus, a typical joystick requires 2 independent buttons and a two-way joystick requires one button.

C. Subtract the number of joystick inputs from the number of rows. 16 rows minus 2 joysticks x 2 inputs = 16 - 4 = 12 remaining inputs. These are available for buttons. Thus, our example will support 2 joysticks and 6 buttons per player. This is perfect for the Street Fighter / Mortal Kombat and almost all MAME games. Also, 16 - 3 joysticks x 2 inputs = 16 - 6 = 10 remaining inputs, so our keyboard will also support 3 joysticks and 3 buttons per player. And 16 - 4 joysticks x 2 inputs = 16 - 8 = 8 remaining inputs, so our keyboard will also support 4 joysticks and 2 buttons per player.

D. What if your results don't match up, e.g., you have more or less rows than the number of desired inputs? I will use some extreme examples to show your options:

1. I want 2 joysticks with 6 buttons each, but my keyboard uses a 12x12 matrix. What can I do?

    a. If you play games that actually require 2 joysticks and 6 buttons per player (SF2), this will not work for you, regardless. Don't even try it! Start over with a different keyboard. The good news is that keyboards using a 16x8 matrix are common and inexpensive.

    b. If you mainly play classic games, it would still be a good idea to try a different keyboard, but you have other options. A 12x12 matrix will support 2 joysticks and 4 buttons per player. Therefore, you could assign buttons 5 and 6 for Player One to the same matrix row as the Joystick 2 movement keys and buttons 5 and 6 for Player Two to the same matrix row as the Joystick 1 movement keys. You would have blocking, but only when button 5 or 6 was pressed at the same time as someone was moving the opposite joystick. This would still work acceptably for the majority of MAME games.

2. I have a simple arcade control panel (say, 2 joysticks and 6 buttons total) and my keyboard uses a 16x8 matrix. What do I do with the additional rows?

    a. In this case, your panel requires 10 inputs so you have six inputs remaining. I would assign keys to the six open rows in the following order: 1P Start, Coin Input 1, 2P Start, Pause, Coin Input 2, Escape, and so forth. If 1 (1P Start) and 5 (Coin Input 1) are on the same matrix row, I would pick a key on another row for Coin Input 1 and configure MAME to recognize the change.

E. Quick Verification - At this point, before we get too far into things it is a good idea to verify that your keyboard does not have a pressed-key maximum imposed in the firmware. To do this, perform the following Steps:

1. Re-assemble and connect the keyboard to the computer.

2. Load one of the keyboard testing programs (Keyscan, Ghostkey) listed in the Verification section.

3. Examine the matrix and select one key from each row. Do NOT select two keys from the same row. Try to select keys that are physically near each other (so you can press them with one hand). Using the DR-104 Keyboard Example, I would select the following keys:
D, S, A, L Ctrl, F (mostly in one row on the physical keyboard)
J, K, L, R Alt, Enter (again mostly in one row on the physical keyboard)
KP1, KP2, KP3, KP Enter, (again in a neat row)
Either L Shift or R Shift, and (optionally) L Win and R Win.

4. Extra hands are useful here: Depress and hold each of the keys selected in Step 3. You should see all the keys display. If no more keys register after you get to eight (or six or five or ten?), then your keyboard has a pressed key maximum and will not be able to use all the inputs. You now have the following options.

    a. If you are building a four player panel, you probably will need to start over again with a different keyboard. (If the limit was eight, then four players activating the diagonals is eight inputs, without any buttons being pressed.)

    b. If you are building a two-player panel, you will probably also want to start over with a different keyboard, but it is not nearly as critical. (If the limit was eight, two players could each activate the diagonals and up to two buttons simultaneously, but no more. This is not acceptable for fighting games, but probably acceptable for most classics.) NOTE: If you choose to use the keyboard anyway, you still want to map out the keys to use as shown below to ensure you don't encounter ghosting/blocking problems.

4. **Choosing the keys:**

You can now assign keys to your MAME inputs. The critical point here is not to put any keys that you don't want to have blocking problems with on the same ROW of the matrix. Keys may be in any column as long as they are not in the same row. Joystick left and right or up and down keys may be in the same row, since they will never be pressed at the same time. As long as these guidelines are followed, any key choice will work and the following steps are just suggestions.

I recommend highlighting the keys as shown below so you can see which keys have already been used or can't be used. Colors are only suggestions and to help you follow along with the appendix A and B examples.

It may be useful to print the matrix out or this section of the instructions to select these keys.

A. There are certain keys which send extra commands to the keyboard buffer and should be avoided. Highlight these non-recommended keys in Gray (for example) as follows: **Direction Arrows** (note that both HotRod and X-Arcade avoid these), Windows Menu Key, L Windows GUI, R Windows GUI, R Ctrl, R Alt, Insert, Home, Page Up, Delete, End, Page Down, PrntScrn, Pause, Keypad Slash, and Keypad Enter. Details of how I came up with this list are available here. Most keys send three characters to the keyboard buffer. These all send five or more. There are a couple of other points to consider below:

- The Power, Wake, Sleep, and Windows Multi-Media keys all fall in the category above, but MAME can't use them, so there is not much point even mapping them.

- While I recommend avoiding these keys even for dedicated encoders, the problem is orders of magnitude worse with a keyboard hack. Dedicated encoders have been optimized for speed and throughput, because they expect to be used with arcade games. Keyboards are designed for typing in a word processor.

- There is a catch-22 with the arrow keys. While MAME itself can be easily programmed to get around using them, many Windows programs and probably even some front-ends for MAME cannot. However, the HotRod is non-programmable, and it does not use them. Furthermore, the odds of a keyboard having the Up and Down arrows on one row and also having the Left and Right arrows on the same row but different from the Up and Down row is pretty slim, so you most likely can't use them even if you wanted to.

- In some cases, you may have no option other than to use these keys. (For example, a matrix row does not contain any keys that aren't on the "Avoid" list.) This is acceptable, the keys will still work, they just slow down the processing, so their use should be minimized.

- The windows GUI keys deserve special consideration. These were not useable in older versions of MAME and are on the "Avoid" list above. I did not use them in my example, because when I wrote this up, MAME could not use them. I did not add them in when I revised the page because the 16x8 matrix without them was representative of most keyboards. However, MAME from version 0.60 up will recognize them and I would gain two additional inputs by using them with the DR-104 keyboard. OTOH, you can't just use the Windows Key in MAME because if you just set the FIRE button to L WIN (for example), MAME will fire a shot and freeze as the Start menu pops up, fire a shot and continue as the Start menu disappears, fire a shot and freeze . . . etc. You can avoid this by installing either WinKey Killer 1.7 (homepage/download) or LogoLess. Here is a comparison of the two programs (either one of which I can recommend):

  WinKey Killer 1.7 - Very simple (double-click to run, double-click again to exit). Transparent. Can be loaded at startup by placing in a shortcut and adding to the Windows Start folder. Kills Windows key instantly in all apps. Has several advanced set-up options like Disable Ctrl+Esc, Disable WinKey+E to launch Internet Explorer, Disable Ctrl-Alt-Del, Hide from Ctrl-Alt-Del, Display message on shutdown, etc.

  LogoLess - Runs with an Icon in the Systray. This is the better program if you ever use the Windows key because it only eliminates it in windows beginning with "MAME" (user-specified), or in full-screen applications.

B. Map the non-gameplay keys. These keys will be allowed to block or mask each other so any matrix position is acceptable (unless you have additional rows available and can make them non-ghosting.) Find the keys you want to use on the matrix and change the shading to yellow (for example) to highlight them. (This prevents you from selecting these keys for gameplay and then finding out you need them for MAME functions). (It's not a bad idea to highlight ALL the keys listed below, even if you don't plan to use them on your panel. This will remind you to change their function in MAME if you decide to use them for action keys when you start assigning the action keys). The following is a list of maintenance keys that I have seen on people's cabinets and my personal suggestions:

   - 1 - For 1P Start (recommended)

   - 5 - For Coin 1 (recommended)

   - 2 - For 2P Start (recommended, unless you never play 2-player games)

   - 6 - For Coin 2 (or map another button such as P1B6 to Coin 2 in MAME. Coin 1 will work for either player except for a few games (Gauntlet, etc)).

   - 3, 4 - For 3P and 4P Start (not recommended unless doing a four player panel)

   - 7, 8 - For Coin 3 and Coin 4 (same as above)

   - P - For Pause (recommended)

   - Esc - For Quit (acceptable or recommend mapping UI CANCEL = 1 AND 2 in MAME.)

   - Tilde - MAME On Screen Display Menu (recommend using a keyboard to adjust this during setup)

   - Tab - MAME Configuration Menu (same as above, however this key is required to bring up the joystick calibration menu. Many MAME games will not play correctly with an out-of-calibration gameport joystick, even though the game may not require it. I recommend having access to this key on any MAME computer which will have a device connected to the gameport. Personally, I recommend mapping CONFIG MENU = 1 AND 5 to provide access to this menu. Then pressing the 1P Start and Coin Input buttons will access this menu.)

- Enter - Used for accepting menu changes or for calibrating analog joysticks/pedals. (Recommend including this under the same circumstance as above. Many people have recommended setting one of the less used buttons (say P2B6) to Enter. This is acceptable or you can simply configure UI SELECT to use the same key as P2B6 is assigned to. The only drawback to either method is that if you are playing a game that requires P2B6 and you do not pause the game before entering the menu, the game may continue to carry out the P2B6 action while you calibrate, etc.!!).
- F12 - Snapshot (not recommended)
- F11 - Toggle speed display (not recommended)
- F10 - Toggle Speed throttling (not recommended)
- F9 - Adjust frameskip (not recommended)
- F8 - Adjust frameskip (not recommended)
- F7 - Load Save State (not recommended)
- F3 - Reset Game (not recommended)
- O and K - To get past "Press OK to continue." See this page for better ways around this. (Not Recommended)

C. If desired (not recommended), try to assign the joystick keys to the arrow keys. This will only be possible if LEFT and RIGHT are on the same matrix row, and UP and DOWN are on the same row, but a different row from LEFT and RIGHT. If this is the case, change the shading for these key to a contrasting color, say teal. Most likely, this will not be possible.

   If this is not possible try to assign the Joystick 1 keys to the numeric keypad arrow keys (2, 4, 6, and 8, especially 2 and 8 which are the down and up keys). These are the HotRod and X-Arcade defaults. If any of these keys can work, shade them in teal. If you can select KP 2 and KP 8, then you can tell EmuLoader (or any frontend with HotRod or X-Arcade support) that you are using these controllers and use the P1 Joystick to scroll the games list. Interestingly, 4 of the 5 keyboards tested allow this and the 5th one doesn't have either key even listed on the matrix.

D. Try to assign the default Player 1 Keys to buttons as follows:

   1. Locate the L CTRL key, if no key on this row is shaded teal, change the shading to aqua. If L Ctrl is on the same row as a teal key, decide whether you would rather have the joystick keys remain or be able to use L Ctrl, and change the shading accordingly. Do NOT leave both L Ctrl and the joystick keys shaded in the same row.
   2. Repeat for L Alt, Space, L Shift, Z, and X. However, if a row already has a key selected, skip it and move on.
   3. You should have most of the Player 1 keys available and highlighted. If any keys were skipped, select replacement keys from any remaining open rows. At this point, you should have six rows with one aqua key in each one highlighted.

E. Assign the joystick inputs as follows: Look through all of the rows with no teal or aqua shading. Either look for logical assignments like Up = U, Down = D, Left = L, Right = R or look for keys which are in the same row and physically oriented like up and down or side by side, or don't worry about it and just pick two keys on each row. Shade the P1 Joystick keys in teal and the P2 Joystick keys in green. When you finish, you should have six rows with one aqua key in each one highlighted, two rows with two teal keys in each highlighted, and two rows with two green keys in each highlighted (ten rows used). The teal and green keys will represent:

   - J1 UP and J1 DOWN
   - J1 LEFT and J1 RIGHT
   - J2 UP and J2 DOWN

- J2 LEFT and J2 RIGHT

F. Assign the remaining rows to P2 button inputs as follows: In any row that does not have any teal or aqua or green shaded keys, pick one key and shade it lime for six of the rows. It is better to pick letter keys than commas or periods (which will be hard to see on the MAME configuration menu) but it really doesn't matter.

G. Double-check your work. At this point, (for a dual joystick panel) you should have six rows with one aqua key in each one highlighted, two rows with two teal keys in each highlighted, two rows with two green keys in each highlighted, and six rows with one lime key in each one highlighted (16 total rows used). If this is not the case, make corrections. At this point, you now know which of the keys you will be using in MAME.

H. You still need to decide which keys will be assigned to which joystick motion or button, but this is basically personal preference as MAME will accept any choice.

5. **Advanced Concepts in Key Assignments (Optional)**

The following information is for the advanced user and is not necessary for general keyboard hacking. If you want the simple approach, proceed to the Verification Paragraph. I felt that I should include this for those that happen to discover it or that want to know the details. While testing the PC Concepts keyboard, I discovered that the "one key per row except the joystick keys" method that I recommend is actually a very useful oversimplification. It is possible to have two (or more) gameplay keys on the same row with no ghosting/blocking provided that no other gameplay keys are in the same **column** as these keys. The following idealized example (which would never work with a real keyboard) will illustrate how this works:

Let's say that you have an 18x6 keyboard matrix and all 14 action keys are mapped to column one and the four joystick keys are mapped between columns one and two so your matrix looks like this:

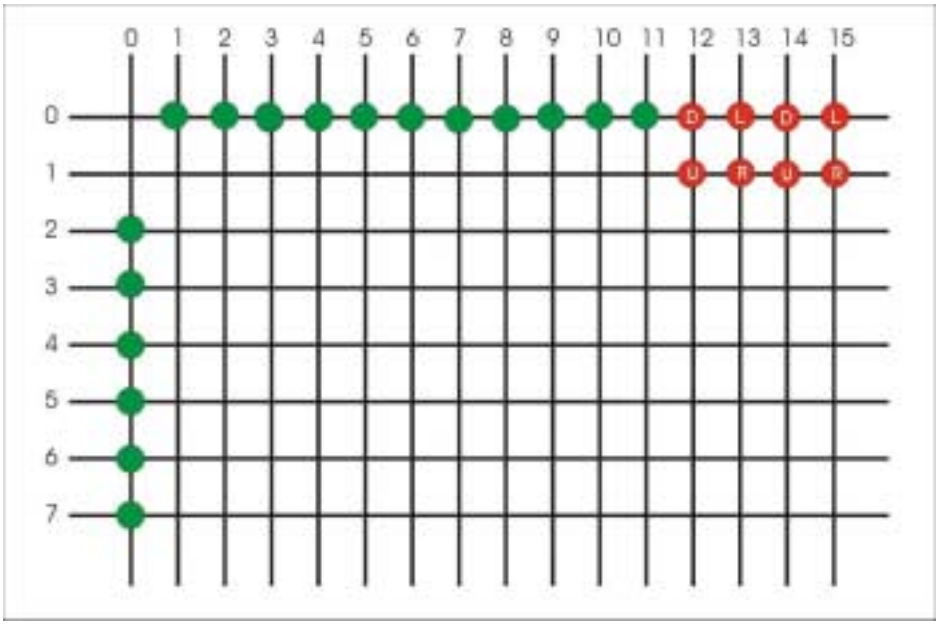
	19	20	21	22	23	24
1	1U	1D				
2	1L	1R				
3	2U	2D				
4	2L	2R				
5	X					
6	X					
...	X					
18	X					

It is possible to move the keys on two of the rows to unused columns and not have ghosting as follows:

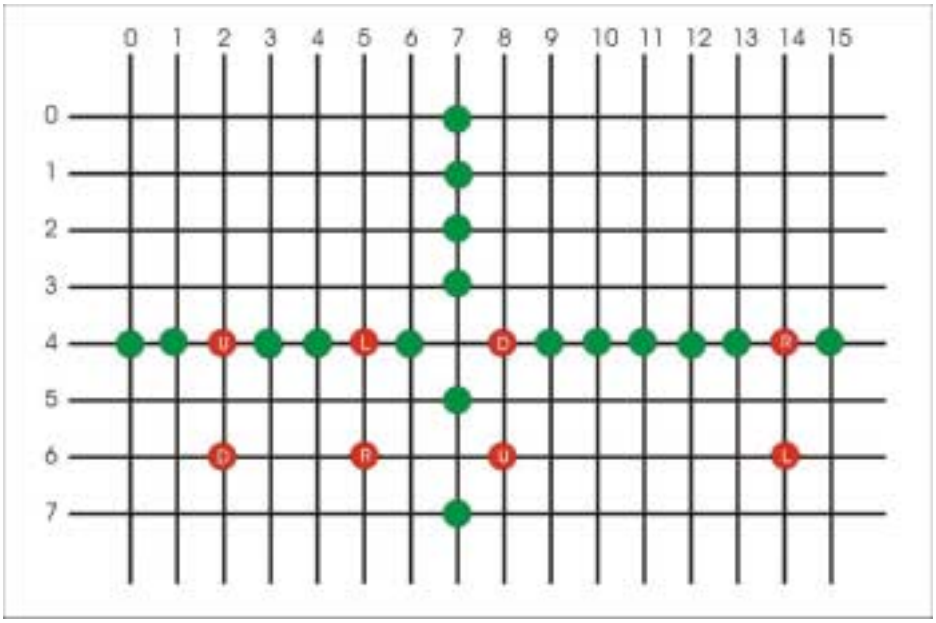
	19	20	21	22	23	24
1	1U	1D				
2	1L	1R				
3	2U	2D				
4	2L	2R				
5			X	X		
6					X	X
...	X					
18	X					

What is the effect of this? The biggest thing is that if there are two keys that you absolutely must have for gameplay that are positioned on the same row, you probably can rearrange your matrix to use these keys. A second advantage is that you may be able to pick up additional available keys using this method. However, in the idealized method above, we only gained two keys. In point of fact, most keyboards will have too many "open spaces" in the matrix to allow keys to be added. In the "real world", you are probably better planning things out using the "one key per row except joysticks" method and only using this method to tweak the matrix if you absolutely want to have some keys on the same row available for gameplay. For example, using this method with the PC Concepts keyboard matrix (see the Appendix), it would be possible to leave the joystick mapped to left and right and also add the L Alt key which is the MAME default for P1B2. However, to do this, you would no longer be able to use the R, Z, X, C, Up, or Page Down keys for gameplay. Each of these keys would have to be moved to a different key on the same row, resulting in a lot of less desirable keys, such as the numeric keypad or function keys.

Here is an even more extreme example of this provided by RandyT (KeyWiz).



This example uses a 16 x 8 matrix and provides 21 inputs, or 25 inputs if you count the joystick inputs which we allow to ghost. The formula for theoretical max inputs is (# of rows + # of columns - 3) plus any extra inputs from allowing the joystick inputs to ghost. Of course we will probably never find a keyboard with all keys on row one and column one available, but this is not required. Consider the following circuit:



This is actually the same circuit as above although the first image is much simpler to visualize. Again, though, in reality, the Apricot matrix is one of the few to have a column containing inputs for all the rows except one, and the row that intersects that column has four inputs unused, so in that better than average example, we would only gain two inputs over the "one per row" method. So this remains mainly a challenge if you get bored or want to see if you REALLY know what's going on with keyboard matrices :-).

6. **Verification:**

   A. Reassemble the keyboard and plug it into the computer.

   B. Download one of the keyboard testing programs from Saint's BYOAC site. A brief description of each program is given here:

- **Keyscan -** When this program starts, a screen appears with a picture of a keyboard and all of the keys grayed out, as a key is depressed, the key on the keyboard display turns black. In addition, a counter on the screen keeps track of how many keys have been depressed and the program beeps if key blocking is encountered until the blocking condition is removed. One drawback to the program is that it will not tell the difference between L Shift and R Shift, L Crtl and R Ctrl, and L Alt and R ALT and will display both as depressed although it only counts one key. Also, the program can not distinguish between the Enter key and the Keypad Enter key and will show both of these as depressed and count two keypresses when either one is depressed. If you are using these keys for MAME and want to be sure of what inputs are being sent to the computer, you will need a different program.

- **Keyjammin and Ghostkey** - To test a keyboard using keyjammin, run the program and press F2. Keyjammin displays all of the keys on a screen in the middle of the window. The keys are not necessarily added in the order in which they are depressed. Keyjammin can tell the L Shift and R Shift keys apart, but not the Ctrl and Alt keys. Ghostkey can recognize any of the keys, but it displays scan codes for each key, which makes the display confusing to read. Also, Ghostkey does not appear to be able to distinguish between the Numeric Keypad Keys and their normal equivalents.

- **Keyhook** - Keyhook displays the last key depressed and continues showing the display even after the key is released. It does not detect multiple keypresses. It is mainly usable for verifying that the control panel operates as intended once the arcade joystick control panel has been assembled.

- **Virtual Keyboard** - This program requires Visual Basic Runtime files. Even though I have installed these, the program still would not run due to the lack of some ActiveX component.

C. A second set of hands is very useful here. You should have 16 keys (in our example) depressed for each of the following steps. Start by pressing each of the twelve button inputs. Now add the keys associated with each Joystick position as follows:

    1. J1 Up, J1 Right, J2 Up, J2 Right
    2. J1 Down, J1 Right, J2 Up, J2 Right
    3. J1 Down, J1 Left, J2 Up, J2 Right
    4. J1 Up, J1 Left, J2 Up, J2 Right
    5. J1 Up, J1 Right, J2 Down, J2 Right
    6. J1 Down, J1 Right, J2 Down, J2 Right
    7. J1 Down, J1 Left, J2 Down, J2 Right
    8. J1 Up, J1 Left, J2 Down, J2 Right
    9. J1 Up, J1 Right, J2 Down, J2 Left
    10. J1 Down, J1 Right, J2 Down, J2 Left
    11. J1 Down, J1 Left, J2 Down, J2 Left
    12. J1 Up, J1 Left, J2 Down, J2 Left
    13. J1 Up, J1 Right, J2 Up, J2 Left
    14. J1 Down, J1 Right, J2 Up, J2 Left
    15. J1 Down, J1 Left, J2 Up, J2 Left
    16. J1 Up, J1 Left, J2 Up, J2 Left

D. If any of these keys do not work, make adjustments to the matrix to fix the problem.

7. **Three and Four-Player Set-Up (Optional)**

This section explains how to expand the keyboard matrix assignments to support 3-player, 3-button games and 4-player 2-button games. A view of how this eventually winds up is available in Appendices A2 and B2. This is an optional step. Paragraph A explains the concept, Paragraph B give the overall theory (in broad terms), and Paragraph C shows how to implement this using the DR-104 keyboard in Appendix A2. This also assumes you have at least a 16-row keyboard matrix.

A. Considerations - To support these games and support a Street Fighter layout without shared inputs would require 25-non ghosting inputs (2 per joystick x 4 = 8, 4 sets of buttons 1 and 2 = 8, Buttons 3 through 6 for Players 1 and 3 = 8, and Player 3 Button 3 = 1). Obviously, there probably is no keyboard that supports this many inputs, so we have the following options:

1. If we will be using a dedicated 4-player cabinet and CP, (for example, we only want to play TMNT, Gauntlet, Simpsons, and similar style games), we can just expand and modify the concepts above, for example, starting from a blank matrix:

    a. Highlight the non-recommended keys (above) in Gray (for example).

    b. Highlight the non-gameplay keys (above) in yellow.

    c. Pick out eight rows and select two keys from each row for the joystick inputs. Shade the P1 Joystick keys teal, the P2 Joystick keys green, The P3 Joystick keys red, and the P4 Joystick keys purple.

    d. Pick out one key from each row of the remaining eight rows for the Button 1 and Button 2 keys. Shade the two Player 1 keys aqua, the two Player 2 keys lime, the two Player 3 keys pink, and the two Player 4 keys violet.

2. If you want to be able to play both types of games on one control panel (two joysticks with six buttons each, one joystick with three buttons, one joystick with two buttons), you will need to share inputs to make this work. This is the example I am referring to in Paragraphs B and C below. Note that if you don't have enough inputs on a row, you could use the same input for one of the Player 3 or 4 inputs and one of the Player 1 or 2 buttons 3 through 6. Note also that while the nature of keyboard hacks does not require keys to share the same inputs, you still will have ghosting/blocking problems if Player 1 or 2 buttons 3 through 6 are pressed during 3-player or 4-player games.

3. The best option for playing both types of games is to use swappable 2-player and 4-player control panels. This allows you to get full usage out of your control panel and also not have to worry about ghosting/blocking. If you use this method, you can either map the 4-player controls completely independently of the 2-player controls, or use the method below which keeps Player 1 using the same controls in either 2-player or 4-player mode.

B. Theory - There are several possible combinations you can use. I will try to make this very general in this section so you can adapt it to your keyboard. I will refer to these as Method A and Method B. Method A is what I used in the DR-104 key example, but either method can work depending on the particular keyboard matrix.

1. Both methods - In both methods, the action buttons for Player 4 (Buttons 1 and 2) are assigned to the same row as the Player 1 and Player 2 Buttons 3. Also, Player 3 Button 3 is assigned to one of the rows that the Player 4 joystick keys use.

2. Method A - In this scenario, the rows containing Player 1 and Player 2 Buttons 5 and 6 are used for the Player 3 and 4 joystick inputs, and the rows containing Player 1 and Player 2 Buttons 4 are used for the action buttons for Player 3 (Buttons 1 and 2).

3. Method B - In this scenario, the rows containing Player 1 and Player 2 Buttons 4 and 5 are used for the Player 3 and 4 joystick inputs, and the rows containing Player 1 and Player 2 Buttons 6 are used for the action buttons for Player 3 (Buttons 1 and 2).

4. Other concerns - In some cases, there may not be enough free spaces on a row to support the desired inputs. This can be resolved either by letting the Player 3 or 4 input use the **same** input as Player 1 or 2, or by re-assigning the Player 1 or Player 2 buttons to place rows with very few keys as Button 1 or Button 2.

C. Implementation - This is a simple step-by-step implementation of this principle using Appendix A2 and the DR-104 keyboard.

1. Find the keys you have assigned for Player 1 and Player 2 Buttons 3, and choose one additional key in each row. These keys will be used as Player 4 buttons 1 and 2. Shade the keys in violet. (KP 1 and I in Appendix A2.)

2. Find the keys you have assigned for Player 1 and Player 2 Buttons 5 and 6, and choose two additional keys in each row. These keys will be used as the Player 3 and Player 4 joystick directionals. Preferably, assign the Joystick 4 inputs to a row with another additional key available (for Player 3 Button 3). Shade the Player 3 keys in red (A,Q and S,W in Appendix A2) and the Player 4 keys in purple (Backspace, Enter and [, 0 in Appendix A2).

   NOTE: Enter is one of the yellow coded keys, but I chose it so it would be easy to see the keys when glancing at the appendix. If I did not want to use this key, I could have used the \ key and let the Player 4 joystick share an input with P2 Button 6.

3. On one of the rows used by the Player 4 joystick directionals, assign one key as Player 3 Button 3 and shade it pink. (Minus key in Appendix A2)

4. Find the keys you have assigned for Player 1 and Player 2 Buttons 4 and choose one additional key in each row and shade it pink. These keys will be Player 3 buttons 1 and 2. (R Shift and Period in Appendix A2).

8. **Modification:**

I have not performed any of the following steps on my own yet. Helpful comments will be greatly appreciated.

A. Separate the conductors from the IDE ribbon cable.

B. For each conductor, strip about ¼ inch of insulation from each wire.

C. If using stranded IDE ribbon cable, tin each conductor using the soldering iron.

D. Disassemble the keyboard and remove the circuit card or circuit card/Mylar sheet(s) assembly.

E. Solder the wires to the circuit card as follows: I recommend soldering to each terminal, even if some will not be used so that if you decide to use a different key assignment, you don't have to resolder the circuit card.

1. If the circuit card is not attached to the Mylar, you can solder directly to the traces on the circuit card.

2. If the Mylar sheet is clamped to the circuit card, you have the following options:

   a. Cut the Mylar below the circuit card and solder to the traces on the Mylar.

   b. Break the clamp over the Mylar sheet and solder to the staples holding the clamp to the circuit card.

   c. Desolder the clamp from the circuit board and solder the wires to the traces on the circuit card itself.

9. **Installation:**

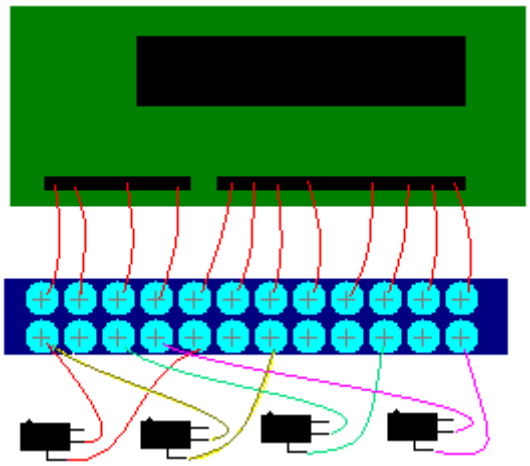
A. Mount the circuit board to the cabinet. Use screws if the board has holes for them. Otherwise, you can use double-sided foam tape. (Thanks, Andy Warne, for this suggestion!)

B. I recommend running all of the wires to a terminal block for easy connection to the buttons/joysticks.

C. For each button input, run one of the corresponding terminal wires to the NO (Normally Open) terminal of the button and one terminal wire to the ground terminal of the button.

For example, for Button 1 (L Ctrl, terminals 4 and 25, using the DR-104 Key) connect the terminal 4 wire to either the NO or ground terminal of the button and the terminal 25 wire to the other terminal of the button.

D. If a terminal connects to more than one button, you may "Chain" the wires together, rather than running separate wires to each button. Below is a graphic provided by tmasman of BYOAC. It doesn't show "chaining" the input wires, but will give you an idea of how to wire the switches to the keyboard hack. (Note that you wire between NO and GND or COM, which might or might not match the terminal locations in the graphic.)



E. Connect the joystick switches to the appropriate terminals. Remember that the joystick "UP" switch is physically down when installed and the same is true for left and right.

F. Run MAME and configure it to recognize the new switches. As you change key assignments either set up ctrl.ini files (MAME 0.60 and up) or, I recommend that you make frequent backups of the C:\MAME\CFG\default.cfg file. This will save you from the following situation, which I encountered.

You assign keys in MAME by pressing the UI SELECT (normally Enter) key to highlight the key to be changed and then typing the new key assignment. If you make a mistake, pressing any key twice will either restore the default setting, or set the key to "None" (I think depending on whether the key was highlighted at the start?) Anyway, while changing the settings for UI SELECT, I made a mistake and wanted to reset it to the default. Accidentally, I set it to "None". Now I was in a Catch-22, because the only way to change the key assignment for UI SELECT away from "None" was to place the cursor over UI SELECT and press the UI SELECT key, which MAME now did not recognize!!! I could not access any function on the Tab menu and every MAME game was affected. After about ten minutes, after I tried deleting the C:\MAME\mame.cfg file and the .cfg file for the particular game in play, I found and deleted the default.cfg file from the C:\MAME\CFG directory and this put everything to rights again. I did not have a backup of this file before I started so I lost all my custom settings, but fortunately, I hadn't changed most of them. Don't let this happen to you!

G. Plug it in and enjoy.

SECTION B - USING ONE OF THE TESTED KEYBOARDS

If you have a keyboard that has been mapped already and you are satisfied with my key assignments as shown in the Appendix, feel free to use it for your design. However, the keyboard manufacturer may have changed the type or revision of the microprocessor between the manufacture of my keyboard and yours, and they would have no reason to tell anyone if they did, so please perform the following before wiring up the keyboard:

1. Plug the keyboard into the computer and run the test programs as shown in Step 5. and verify that all of the same keys work without blocking on your keyboard.

2. Disassemble the keyboard and verify that each key that you will use on your control panel is mapped to the same terminals using the methods in Step 2.

FAQ's

1. Is there an easier way to do this?

   Not really, at least not to get the full functionality of the keyboard. For a simple (one joystick, two buttons) control panel, you probably could just run keyscan and find six keys that work and then trace only these keys out. The methods shown here are not that difficult though. The procedure seems long because I went into extra detail to make it easy to follow (hopefully).

2. What can I do to help?

   Two things, if you build a keyboard hack, send in the keyboard construction specs and the keyboard matrix, so we can add it to the appropriate sections of the page. Also, if you find any errors or helpful suggestions, E-mail me.

3. Are diodes useful?

   As stated in the main text, diodes positively will NOT help if your keyboard has a simultaneous pressed-key limit or key blocking problems. (And all the keyboards I tested had one or the other.). If your keyboard has classic ghosting (you press three keys simultaneously and a fourth key also registers), then diodes SHOULD allow any keypress to register without ghosting. However, you can make any keyboard work without diodes using the methods above.

4. If my keyboard has classic ghosting (see above), how would I use diodes to help?

   The following is based on my research on keyboard encoders and my memory of ancient BYOAC message posts, so it may not be entirely accurate, but I think it will be close. Again, I will use the DR-104 keyboard and Appendix A as an example, even though this keyboard would NOT benefit from the use of diodes. Let's consider the F key (Terminals 5 and 21). It is only necessary to use diodes on the row or the column of the matrix, not both. Since there are less columns, that is the logical place to attach them. So, we would connect either the NO or COM terminal of a switch to Terminal 5 of the keyboard hack, and the other switch terminal to a diode before connecting to Terminal 21 of the keyboard. Diodes have polarity (there will be a line on the diode near one of the ends), so it matters which direction the diode is connected, but the "correct" connection will vary from keyboard to keyboard. Here's the test, though: Wire the circuit as above, connect to the PC and open Notepad. Close the switch. If "F" displays, you're set, if nothing displays, reverse the diode.

   Things I **don't** know: Assuming Column 21 and Key F wants the diode positioned to allow current to flow "From" the switch "To" the keyboard hack. I don't know if Columns 22 through 26 will then want current to flow "From" the switch also? And let's say I want to connect the A key along with the F key. I don't know if I will need separate diodes for A and F (probably), or if I can just connect the A and F switch wires together before they go through the diode?

Appendices

Appendix A, A2, B, and B2 below have been evaluated by me and are included as examples of what this method can accomplish. All subsequent matrices are included for reference only. No responsibility is assumed for any errors that may be found on any of these matrices.

Appendix A – Digital Research DR-104Key Example Keyboard Matrix

Looking at component side of circuit card, terminals 1-26, 1 is on same side as PC cable connector, moving left to right across the bottom edge of the keyboard encoder circuit card.

	Maintenance	P1 Joystick	P1 Button	P2 Joystick	P2 Button	Unusable		
	19	20	21	22	23	24	25	26
1	E	F3	D	F4	C		F2	3
2	W	Caps Lock	S		X		F1	2
3	Q	Tab	A	Esc	Z		Tilde	1
4	Break				R Ctrl		L Ctrl	F5
5	R	T	F	G	V	B	5	4
6	U	Y	J	H	M	N	6	7
7	I	]	K	F6	Comma		=	8
8	O	F7	L		Period	Win Menu	F8	9
9	P	[	Semicolon	Apostrophe		/	-	0
10	Scroll Lock			L Alt		R Alt		Print Screen
11		Backspace	\	F11	Enter	F12	F9	F10
12	KP 7	KP 4	KP 1	Space	Num Lock	Down	Delete	
13	KP 8	KP 5	KP 2	KP 0	KP /	Right	Insert	
14	KP 9	KP 6	KP 3	KP Period	KP *	KP -	Page Up	Page Down
15	KP +		KP Enter	Up		Left	Home	End
16		L Shift	R Shift					
17		L Win						
18			R Win					
	19	20	21	22	23	24	25	26

P1 Up - KP 8	P2 Up - C
P1 Down - KP 2	P2 Down - D
P1 Left - KP 3	P2 Left - F
P1 Right - KP 6	P2 Right - G
P1 B1 - L Ctrl	P2 B1 - KP +
P1 B2 - L Alt	P2 B2 - M
P1 B3 - Space	P2 B3 -]
P1 B4 - L Shift	P2 B4 - L
P1 B5 - Z	P2 B5 - /
P1 B6 - X	P2 B6 - \

APPENDIX A2 - Digital Research DR-104Key 4-Player Example Keyboard Matrix

Enter is normally a reserved yellow key, but it was a logical choice for the P4 Joystick.

	Maintenance	P1 Joystick	P1 Button	P2 Joystick	P2 Button	P3 Joystick	P3 Button	Unusable	
	P4 Joystick	P4 Button							
	19	20	21	22	23	24	25	26	
1	E	F3	D	F4	C		F2	3	1
2	W	Caps Lock	S		X		F1	2	2
3	Q	Tab	A	Esc	Z		Tilde	1	3
4	Break				R Ctrl		L Ctrl	F5	4
5	R	T	F	G	V	B	5	4	5
6	U	Y	J	H	M	N	6	7	6
7	I	]	K	F6	Comma		=	8	7
8	O	F7	L		Period	Win Menu	F8	9	8
9	P	[	Semicolon	Apostrophe		/	-	0	9
10	Scroll Lock			L Alt		R Alt		Print Screen	10
11		Backspace	\	F11	Enter	F12	F9	F10	11
12	KP 7	KP 4	KP 1	Space	Num Lock	Down	Delete		12
13	KP 8	KP 5	KP 2	KP 0	KP /	Right	Insert		13
14	KP 9	KP 6	KP 3	KP Period	KP *	KP -	Page Up	Page Down	14
15	KP +		KP Enter	Up		Left	Home	End	15
16		L Shift	R Shift						16
17		L Win							17
18			R Win						18
	19	20	21	22	23	24	25	26	

P1 Up - KP 8	P2 Up - C	P3 Up - W	P4 Up - Backspace
P1 Down - KP 2	P2 Down - D	P3 Down - S	P4 Down - Enter
P1 Left - KP 3	P2 Left - F	P3 Left - A	P4 Left - [
P1 Right - KP 6	P2 Right - G	P3 Right - Q	P4 Right - 0
P1 B1 - L Ctrl	P2 B1 - KP +	P3 B1 - R Shift	P4 B1 - KP 1
P1 B2 - L Alt	P2 B2 - M	P3 B2 - Period	P4 B2 - I
P1 B3 - Space	P2 B3 -]	P3 B3 - -	
P1 B4 - L Shift	P2 B4 - L		
P1 B5 - Z	P2 B5 - /		
P1 B6 - X	P2 B6 - \		

Appendix B – PC Concepts KWD-203 Example Keyboard Matrix

Looking at component side of circuit card, terminals 1 - 18, right to left across top edge of circuit card; terminals 19-27, top to bottom along left edge of circuit card. The upper Mylar sheet goes to row terminals 1-18 and the bottom Mylar goes to column terminals 19-27.

R Ctrl and End are normally unusable keys, but were the only ones available in these rows.

	Maintenance	P1 Joystick	P1 Button	P2 Joystick	P2 Button	Unusable				
	19	20	21	22	23	24	25	26	27	
1	5	T	G	B	Tab	Space	Caps Lock	Tilde		1
2									R Win	2
3									R Ctrl	3
4	1	Q	A	Z	KP Period	KP 0	KP +	KP -		4
5		L Alt		Left	Right		R Alt	Down		5
6			R Shift	Apostrophe	Enter	L Shift		KP Enter		6
7	]	[		\	0	P	Semicolon	/		7
8	2	W	S	X	KP 3	KP 6	KP 9	KP +		8
9	3	E	D	C	KP 2	KP 5	KP 8	KP /		9
10									L Win	10
11									L Ctrl	11
12	4	R	F	V	KP 1	KP 4	KP 7	Num Lock		12
13	Win Menu		End		Page Dn	Home	Insert	Page Up		13
14			Break	Up	Hyphen	Scroll Lock		Delete		14
15	F11	F12	=	Backspace	9	O	L	Period		15
16	F7	F10	F9	F8	8	I	K	Comma		16
17	F3	F6	F5	F4	7	U	J	M		17
18	Esc	F1	F2	Print Screen	6	Y	H	N		18
	19	20	21	22	23	24	25	26	27	

P1 Up - KP 8	P2 Up - U
P1 Down - KP 2	P2 Down - J
P1 Left - R	P2 Left - Y
P1 Right - F	P2 Right - H
P1 B1 - L Ctrl	P2 B1 - R Ctrl
P1 B2 - L Alt	P2 B2 - End
P1 B3 - Space	P2 B3 - I
P1 B4 - L Shift	P2 B4 - Hyphen
P1 B5 - Z	P2 B5 - L
P1 B6 - X	P2 B6 - /

Appendix B2 – PC Concepts KWD-203 4-Player Example Keyboard Matrix

R Ctrl and End are normally unusable keys, but were the only ones available in these rows.

	Maintenance	P1 Joystick	P1 Button	P2 Joystick	P2 Button	P3 Joystick	P3 Button	Unusable		
	P4 Joystick	P4 Button								
	19	20	21	22	23	24	25	26	27	
1	5	T	G	B	Tab	Space	Caps Lock	Tilde		
2									R Win	1
3									R Ctrl	2
4	1	Q	A	Z	KP Period	KP 0	KP +	KP -		3
5		L Alt		Left	Right		R Alt	Down		4
6			R Shift	Apostrophe	Enter	L Shift		KP Enter		5
7	]	[		\	0	P	Semicolon	/		6
8	2	W	S	X	KP 3	KP 6	KP 9	KP +		7
9	3	E	D	C	KP 2	KP 5	KP 8	KP /		8
10									L Win	9
11									L Ctrl	10
12	4	R	F	V	KP 1	KP 4	KP 7	Num Lock		11
13	Win Menu		End		Page Dn	Home	Insert	Page Up		12
14			Break	Up	Hyphen	Scroll Lock		Delete		13
15	F11	F12	=	Backspace	9	O	L	Period		14
16	F7	F10	F9	F8	8	I	K	Comma		15
17	F3	F6	F5	F4	7	U	J	M		16
18	Esc	F1	F2	Print Screen	6	Y	H	N		17
	19	20	21	22	23	24	25	26	27	

P1 Up - KP 8	P2 Up - U	P3 Up - Q	P4 Up - =
P1 Down - KP 2	P2 Down - J	P3 Down - A	P4 Down - Period
P1 Left - R	P2 Left - Y	P3 Left - W	P4 Left - [
P1 Right - F	P2 Right - H	P3 Right - S	P4 Right -]
P1 B1 - L Ctrl	P2 B1 - R Ctrl	P3 B1 - R Shift	P4 B1 - B
P1 B2 - L Alt	P2 B2 - End	P3 B2 - Scroll Lock	P4 B2 - Comma
P1 B3 - Space	P2 B3 - I	P3 B3 - 0	
P1 B4 - L Shift	P2 B4 - Hyphen		
P1 B5 - Z	P2 B5 - L		
P1 B6 - X	P2 B6 - /		

Appendix C – Apricot Computers RT-6656TUK Example Keyboard Matrix

16 x 8 Matrix. I did not map out or select these keys. 8 was not the ideal choice as it is used for Coin 4 by MAME. Not a problem for a two-player panel, though. Down, Right Control, Insert, Home, and End were poor choices because of buffer concerns, but this info was not part of the article when this keyboard was mapped.

	Maintenance	P1 Joystick	P1 Button	P2 Joystick	P2 Button	Unusable			
	21	22	23	24	25	26	27	28	
1	LED								1
2	LED								2
3	LED								3
4	LED								4
5		Right Ctrl				Left Ctrl			5
6	KP -	KP *	KP 3	KP 9	Page Down	Page Up	KP 6	KP Delete	6
7	Left	Pause	KP Enter		End	Home	KP +	Up	7
8	R Alt			Scroll Lock	Prnt Scrn			Alt	8
9		Shift					Shift		9
10	Right	KP /	KP 2	KP 8	F12	Insert	KP 5	KP 0	10
11	Down	Num Lock	KP 1	KP 7	F11	Delete	KP 4		11
12		<	K	I	8	Plus	]	F6	12
13	Space bar	Enter			F10	F9	Backspace	F5	13
14	?	#	;	P	0	Minus	[	@	14
15		>	L	O	9	F8	F7		15
16	N	M	J	U	7	6	Y	H	16
17	B	V	F	R	4	5	T	G	17
18		C	D	E	3	F2	F3	F4	18
19		X	S	W	2	F1	Caps Lock	\	19
20		Z	A	Q	1	Tilde	Tab	Esc	20
	21	22	23	24	25	26	27	28	

P1 Up - End	P2 Up - T
P1 Down - Home	P2 Down - G
P1 Left - KP -	P2 Left - Y
P1 Right - KP 3	P2 Right - H
P1 B1 - Insert	P2 B1 - Right Ctrl
P1 B2 - Space	P2 B2 - C
P1 B3 - Alt	P2 B3 - Z
P1 B4 - 0	P2 B4 - Shift
P1 B5 - Cursor Down	P2 B5 - L
P1 B6 - 8	P2 B6 - S

Appendix D – Compaq model: SK-2800c, Compaq p/n: 341162-006, Spare p/n: 387084-003 (Thanks Mr. Arcade)

I did not map out or select these keys. NOTE: Space and S should not both have been used. One of them should be swapped with either Num - or Num + in row 4 or 5. F10, F11, and F12 are used by MAME and would have to be re-mapped in MAME. The Arrow keys were poor choices because of buffer concerns, but this info was not part of the article when this keyboard was mapped.

	Maintenance	P1 Joystick	P1 Button	P2 Joystick	P2 Button	Unusable		
	18	19	20	21	22	23	24	25	
1							R Ctrl	L Ctrl	1
2							\	L Shift	2
3	Win-Menu	End			R Alt	L Alt			3
4		KP Period		R Win					4
5	Scr Lock	KP +	L-WIN						5
6	Down	KP -	Prt Screen	KP 1	KP 4	KP 7	F11	Apostrophe	6
7	Right	KP /	KP 0	KP 2	KP 5	KP 8	F12	Semicolon	7
8	Page Down	Enter	Page Up	KP Enter	KP 6	KP 9	F10	F9	8
9	Left	Home	Delete	Insert		P	0	Minus	9
10	KP *	/	\	L	]	O	9	F8	10
11	R Shift	Period	F6	K	F7	I	8	=	11
12	Backspace	Comma	H	J	[	U	7	6	12
13	M	B	G	F	Y	R	4	5	13
14	N	V	F4	D	T	E	3	F2	14
15	Space	C	Z	S	F3	W	2	F1	15
16	F5	X	Esc	A	Tab	Q	1	Tilde	16
17	Num Lock		Up	Caps Lock	Enter			Pause	17
	18	19	20	21	22	23	24	25	

Appendix E – Micro Innovations - model KB400i (Thanks Mr. Arcade)

I did not map out these keys. I have no idea why some keys appear to take two columns, nor why Space, R Shift, and L Shift appear twice in the matrix. I did assign the inputs. Since I can't verify the matrix, I just did not assign L Shift and Space in rows where it would conflict with keys I wanted to use.

	Maintenance	P1 Joystick	P1 Button	P2 Joystick	P2 Button	Unusable			
	19	20	21	22	23	24	25	26	
1	1	Tilde		Z	Esc	A	Tab	Q	1
2	2	F1		X	L Shift	S	Caps Lock	W	2
3	3	F2		C	F4	D	F3	E	3
4	4	5	B	V	G	F	T	R	4
5	7	6	N	M	H	J	Y	U	5
6	8	=		Num Lock	F6	K	]	I	6
7	9	F8	Win Menu	Period		L	F7	O	7
8	0	Minus	/		Apostrophe	Semicolon	[	P	8
9	F10	F9	F12	Enter	F11	\	Backspace		9
10	Print Screen		R Alt	*Power	L Alt			Scr Lock	10
11	Space	Insert	Right	KP /	KP 0	KP 2	KP 5	KP 8	11
12		Delete	Down	Comma	Space	KP 1	KP 4	KP 7	12
13	Page Down	Page Up	KP -	KP *	KP Period	KP 3	KP 6	KP 9	13
14	End	Home	Left	*Sleep	Up	KP Enter		KP +	14
15				*Wake		R Shift	L Shift		15
16				R Shift		R Win			16
17				R Ctrl			L Win		17
18	F5	L Ctrl						Pause	18
	19	20	21	22	23	24	25	26	

Appendix F – Non-Branded Keyboard, chicony ver a 105-06868-010 6868A-0017 9947M FADD3 keyboard encoder (Thanks Pete Clements)

I did not map out the matrix for this. Ghostkey codes are included. The F11 key is missing. It looks like Pete used Ghostkey to map this out, and since it can't map the NumPad keys, most of these are missing. I assigned the inputs. I think Right, Up, Left, Down, Insert, Page Up, Page Down, Delete, End, Home, Enter, Up, Left appear twice in the Matrix because one of them is the KP version, and Ghostkey can't tell them apart. I have tried to avoid these keys when assigning the inputs, just to be safe.

	Maintenance	P1 Joystick	P1 Button	P2 Joystick	P2 Button	Unusable				
	19	20	21	22	23	24	25	26	27	
1	Insert 082	Blank 088	Up 072	KP 5 076	Down 080	Insert 082	KP / 122	Right 077		1
2	F9 067	F10 068	Blank 125	Backspace 014	\ 043	F5 063	Enter 028	Space 057		2
3	Delete 083	Blank 087	Home 071	Left 075	End 079		Num Lock 069	Down 080		3
4	Page Up 073	Page Down 081	Page Up 073	Right 077	Page Down 081	Delete 083	KP * 055	KP - 074		4
5	Home 071	End 079	KP + 078	Blank 126	Enter 028	Up 072	Pause 123	Left 075		5
6		Print Screen 084	Scr Lock 070			L Alt 056		R Alt 121		6
7				L Shift 042			R Shift 054			7
8	L Ctrl 029						R Ctrl 120			8
9	Tilde 041	1 002	Q 016	Tab 015	A 030	Esc	Z 044			9
10	F1 059	2 003	W 017	Caps Lock 058	S 031	F12 086	X 045			10
11	F2 060	3 004	E 018	F3 061	D 032	F4 062	C 046			11
12	5 006	4 005	R 019	T 020	F 033	G 034	V 047	B 048		12
13	6 007	7 008	U 022	Y 021	J 036	H 035	M 050	N 049		13
14	= 013	8 009	I 023	] 027	K 037	F6 064	Comma 051	Blank 115		14
15	F8 066	9 010	O 024	F7 065	L 038		Period 052			15
16	- 012	0 011	P 025	[026	Semicolon 039	Apostrophe 040	\ 043	/ 053		16
17			Win Menu 093	R Alt 093	R Win 092	Blank 112				17
18	L Win 091									18
	19	20	21	22	23	24	25	26	27	