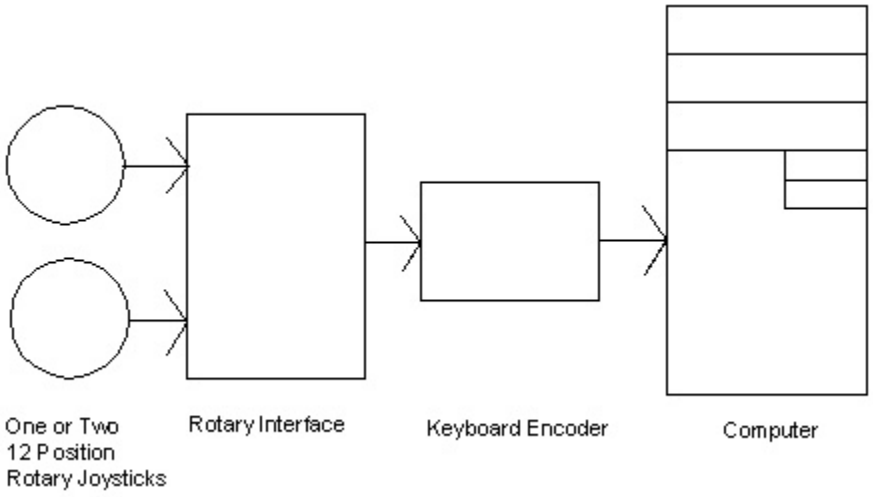
12 Position Mechanical Rotary Joystick Interface

Design and development by Jason Penney
Additional revisions based on ideas contributed by Joel Simpson

***This document assumes that the reader is generally familiar with the intended use of this product, the emulator program it is used with, and the functionality of the other hardware required (input joysticks and output keyboard interfaces, etc). All pictures and text should be clear and legible when this document is printed on paper, if they seem otherwise distorted in a file viewer.

Features

- Converts a 12 position mechanical rotary joystick's 13 pin cable into two automatic output switches per joystick: Clockwise and Counter Clockwise rotation
- Allows 1 or 2 rotary joysticks to be monitored continuously
- More authentic game play than using an *optical* rotary joystick for games designed around the 12 position rotary joystick
- Can simulate an optical rotary joystick's freewheeling spinner type action when the emulator sensitivity and speed settings are adjusted properly, making the mechanical clicks overshoot quickly!
- Avoids the need to waste X and Y axis serial interfaces required by *optical* joysticks
- Output switches are turned on momentarily, then off again each time the joystick shaft rotates clockwise or counter clockwise, just as if a button were physically being pressed and released
- Output switches operate exactly like SPST (2 wire single pole single throw) normally open pushbutton switches (such as standard arcade buttons)
- Straightforward operation: Plug 1 or 2 joystick cables into the interface, wire the output switch terminals anywhere a switch would normally be wired to activate the dial/twist rotate commands on a keyboard encoder, and connect a 5 volt power supply:



Uses

The rotary interface (RI) may be used to allow the special **12 position mechanical** rotary joysticks (not the ***optical*** rotary joysticks, that's a different type of connection) to work with any game that supports two buttons per player for character motion in the clockwise and counter clockwise directions. Such buttons are referred to as the *Dial* or *Twist* buttons in the emulator configuration menu. Once the device is connected and verified as operational, all further adjustments are made from within the emulator itself. The sensitivity and speed of the rotation commands may be individually adjusted in the emulator to tweak each game for optimal performance with this interface.

Description

The objective of this product is to act as a drop in replacement for switches that would have to be used to manually rotate a game character clockwise and counter clockwise. Each time the joystick shaft is rotated, the appropriate player's clockwise or counter clockwise screw terminal pair will be temporarily shorted electrically, simulating a switch being pressed, then released. The keyboard encoder that the RI is wired to then detects the button press and sends the proper keystrokes to the computer.

The interface provides output connections that are electrically equivalent to normal micro switches found in arcade buttons and joysticks. Each output switch equivalent on the interface is connected to the keyboard encoder through a marked pair of screw terminals, which may be wired to any point in the system that an arcade switch would otherwise be wired to activate keystrokes. This could be a keyboard hack or a stand-alone keyboard encoder, operating in either direct or matrix mode. Each marked pair of screw terminals are treated the same way as a pair of contacts on a micro switch that you would wire to a keyboard encoder.

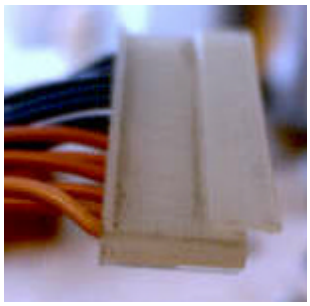
How The RI Acts Like An Arcade Button

In a physical arcade switch (normally open, single pole single throw, which means you have to press the switch to make contact between two wires), there is normally no connection between the two switch contacts. When you press the switch, the circuit path between the two contacts is closed. When an arcade switch is wired to a keyboard encoder, each contact on the switch is wired to a designated pair of inputs on the encoder. This may be a ground connection along with a dedicated input connection, or a row and a column connection of a matrix. From the perspective of a keyboard encoder, all you need to do to activate a keystroke for any given input is take the two wires belonging to that input, either a row and column pair, or a single input and a ground wire, and short them together manually instead of using an arcade switch. The keyboard encoder will respond to either method of closing the circuit path between the two wires to activate the key press. The joystick interface does this electronically. For each pair of screw terminals that are wired to a keyboard input in place of an arcade switch, the two terminals will be shorted together momentarily, when necessary, to activate a rotation keystroke.

The Joystick Cable

The wires on the joystick cable may exist in different color schemes. One known scheme contains a rainbow of colors, while another contains mostly blue and red wires, with a black and orange wire on either end. Both types have the same "R" marking on one end of the cable. **This is the end that must plug into the joystick**.

The other end plugs into the RI board. The wires are connected in a different order on each end of the cable, so the correct orientation is essential for operation. The cable is inserted with the hooded side of the plug facing up, and the exposed metal contacts facing down on both ends of the cable (joystick side and interface board side). When inserted this way on the interface, looking at the plug from behind its wires, the wire on the right of the connector is the ground wire and the other 12 from left to right are the rotary switch positions, in sequential order of rotation.

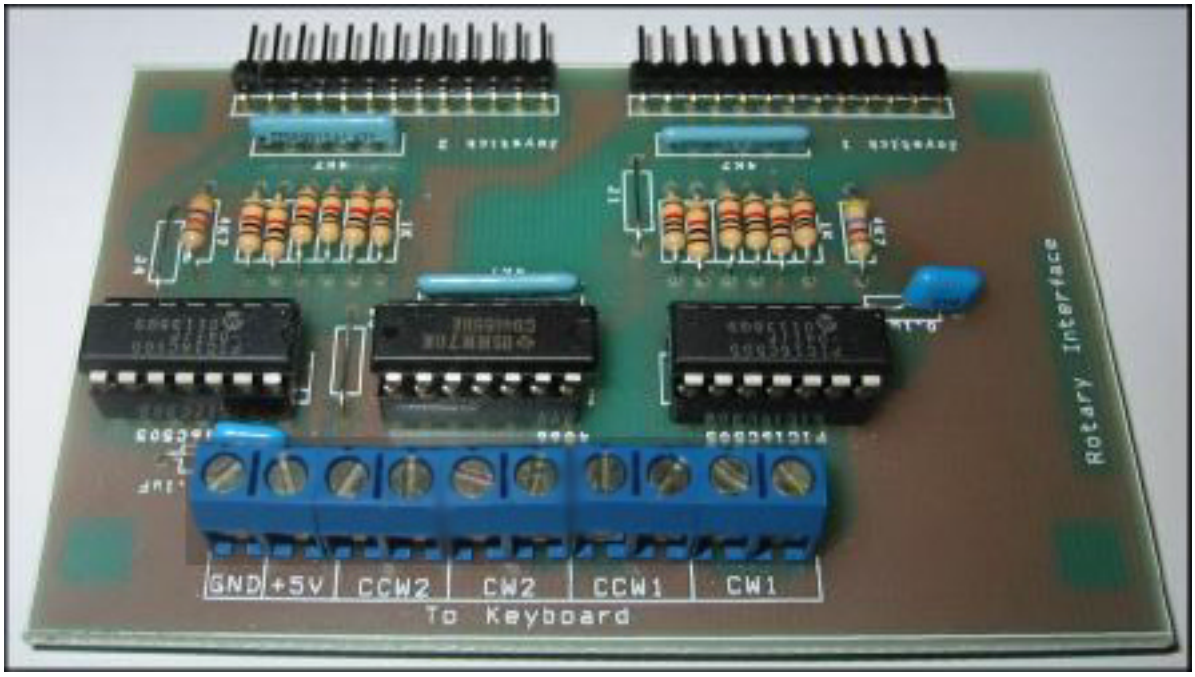


The cables are plugged in with the hood facing up

Setting Up The Interface

To plug the joystick cable into the interface, line up the pins as best as possible and push the connector in. It may be easier to insert the connector with a side-to-side motion, as long as the pins aren't subjected to too much stress.

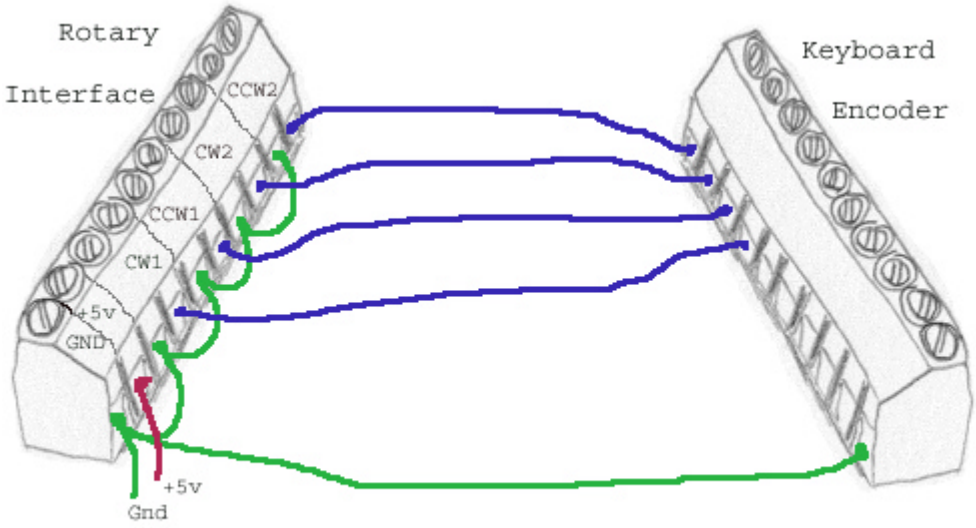
Connecting To A Keyboard Encoder



To use the output screw terminals, since there are two terminals per switch, wiring to a keyboard input is exactly the same as a normal micro switch. Any designated pair of screw terminals on the RI may be wired without regard to polarity (for example, it doesn't matter which of the two terminals becomes a ground wire or which one connects to a row or column). If the keyboard controller is connected in a matrix configuration, either of the two switch terminals for a given switch would connect to the desired "row" wire of a matrix, while the other switch terminal would connect to the "column" wire. When the interface's micro controller sends a "high" activation signal to one of the 4066 IC's control input lines, the switch closes momentarily and that specific row and column get connected together long enough to simulate having pressed a physical switch.

If the keyboard circuit is used in direct mode, where each switch on the keyboard circuit has a dedicated input line and a common single connection, the RI's switch terminals would simply be connected with one terminal at the required input line for each switch, and the other terminal would be connected to the common point, most likely ground.

Shown below is a hand drawn sketch of a simple connection diagram to illustrate how to wire the RI to a keyboard encoder operating in direct mode. The labeling on the RI's screw terminals in this illustration is arbitrary. Each unit will have the proper connection labels attached to its screw terminals, but the general layout will be the same, where each set of switch contacts will be arranged with the two contact screw terminals side by side.



Rotary Interface connected to a keyboard encoder in Direct Mode.

In this example, using direct mode, one terminal out of each pair of switch contacts must be wired straight to the keyboard encoder's input line that you have set aside for a given player's rotary buttons. These connections are shown as blue wires for each switch in the diagram. For two players, four rotary switches are wired to the keyboard encoder: Player 1 CW (clockwise), Player 1 CCW (counter clockwise), Player 2 CW, and Player 2 CCW.

The other switch contact for each screw terminal pair must be connected to the keyboard encoder's common connection for its switch inputs. Since this is most likely the ground of the power supply, this example shows that the other screw contacts are wired to each other, then to the power supply ground of both pieces of hardware. The RI's power supply terminals are connected to the +5 volt supply.

If the RI were to be used in a matrix system, the only difference in the above diagram would be that instead of having the second screw terminal of each pair wired to the common point, they would each separately wire into another input of the keyboard encoder, designated as a row or column, depending.

Power Supply

When there is no power connected to the interface, the switch outputs will behave ambiguously. It is not recommended to connect the switches to another powered circuit (eg. keyboard) while the unit itself is not powered.

When the interface is powered on, the output terminals are fixed in one of two states: completely turned on (switch pressed) or completely turned off (switch released). When the interface is powered down, current may still possibly flow through the screw terminals, so it may end up simulating a key press on all four output switches constantly. It is generally not a good idea to allow current to flow through an integrated circuit chip that is not powered on.

As long as all components are controlled by the same power switch, as in a computer system, everything should be ok. If the interface is powered from a separate AC or DC adapter, it should at least be plugged in somewhere that it will be turned on at the same time, or before the computer is turned on. An example situation involves just using the same power bar for all components of the system, which ensures that the computer can never be powered on before the interface.

The circuit is powered from a +5 volt supply, which may be obtained from another part of the computer, anywhere it can be tapped from. Ideally the keyboard encoder circuit would be the best place to tap power. The screw terminals provide easy hook up to the power source. If power is not directly available and it is not preferable to solder on the keyboard encoder's circuit board, it may be possible to obtain a computer power cable splitter used for hard drives, and cut the connector to obtain exposed wires for +5v and ground. ***It is important to connect the power supply to the proper terminals and in the proper polarity, with ground-to-ground, +5 to +5 to avoid damage to the interface.*** It is recommended to verify the polarity and voltage level with a meter before connecting the power source to the interface.

Initial Functional Test

When at least one joystick has been connected to the interface, and it has been given a power source and wired to a keyboard encoder, the unit may be tested in a game, or simply by loading a text editor and rotating the joystick(s) to determine if the keystrokes are being registered. If there is a problem, troubleshooting of the interface board may be required (see later in document) to ensure that the interface is installed correctly and that it is functioning as intended.

Configuring For Optimal Performance

Some of the games respond differently to the rotary dial/twist buttons than others, and different computer configurations/speeds may also require tweaking to find the best response settings. Some games will seem to respond slowly to rotations, while other games may over-trigger and seem to rotate too many times for each single joystick rotation. To set the proper response characteristics for each game, the configuration menu will have to be accessed for each game, and trial and error will be required to get the character responding properly.

By accessing the configuration menu with the "Tab" key, then the "Analog Controls" menu option, the Dial or Twist buttons for player 1 and player 2 may be fine-tuned individually for each game. The sensitivity and speed may be changed so that each game works well with the dial buttons, whether they are being pressed manually on a keyboard, or automatically with this joystick interface.

Some games were naturally designed to be more responsive to a "dial" than others, having more possible positions for the character to stop at during a full circle rotation. Consider the circular face of a clock with 12 numbers on it. If you manually push a hand from one number to the next, record the amount of time it took you to push the hand between numbers. Then take a circle of the same size and put 50 numbers around the face instead of just 12. Push the hand of the clock for the same amount of time on this new layout, and you will see that you have passed more than just one numerical position in that amount of time.

If you hold the buttons down too long, the character begins to auto-rotate so you must set the sensitivity and speed for the game to react slower when that happens. If the case were such that the button presses seemed too

fast for the games to pick them up every time reliably, you would adjust the dials to make them more sensitive until the rapid button presses could be finally detected on every hit.

The best way to calibrate the buttons is to start a game, move the character somewhere other than the center of the screen (so it is not hidden by the configuration menu that pops up), then enter the dial adjustment menu in the Analog Controls section of the Tab menu. Use the arrow keys to change the dial settings, and then rotate the joystick to see the character's responsiveness to each change until it is optimal. The new settings should automatically be saved for each game.

Once all games have been calibrated to the constant button press time of the rotary interface, they may all be played with optimal results.

It is important to note that even when the games are tweaked to work optimally, there may be instances where the game character doesn't seem to respond properly. These are physical limitations of the emulator software and/or the joystick itself.

One such possibility occurs when the joystick is rotated one position, but the character does not respond, or the character may over-trigger and rotate more than once. This is an issue with the emulator itself, as would be noticed if the dial keys were pressed on the keyboard instead of with the rotating joystick. Keeping in mind that this is an emulator to begin with and is prone to imperfect reproduction of intended actions, if you press a dial key on the keyboard repeatedly, at a fixed rate, eventually you may find that one of your key presses went undetected, or the character double-rotated in that direction. The rotary joystick may thus provide the same result.

Another possible situation occurs when the character rotates as expected when the joystick is rotated, but the character also then goes on to rotate backwards for a single rotation in the wrong direction after the proper rotations take place. This is a limitation of the rotary joystick itself and only occurs at very rapid rotational speeds (such as attempting to use the joystick as a freewheeling spinner, which is not its intended use!) This has been observed only at exaggerated rotational speeds, and hasn't adversely affected normal game play. Overall, the idea is to rotate the joystick at a moderate rate and not to use it as a spinner.

A Note About The Output Switch Closure Timing

The output switch terminals of the RI are configured to be closed (pressed) for a very short duration, and then released. There is no time delay imposed after releasing the switch, other than the time it takes to read in the next joystick status and process it (negligible time frame). What this means is each time the joystick is rotated, if the rotations are occurring fast enough, the on/off cycles will be so fast that it may seem as if the output switch on the RI is being held in the pressed position for that whole rotating duration, instead of being rapidly opened and closed. It may appear to a keyboard encoder that the key is being held constantly down instead of rapidly pressed. This does not pose a problem, and in fact has been found to enhance performance in the emulator. This method of operation for the switches, with minimal delay times, was implemented to allow the joystick switches to be monitored as fast as possible. If delay times were imposed to properly "release" each individual switch press, valuable joystick sampling time would be compromised, and the results would not be as accurate and smooth as they are.

If the joystick is rotated slowly, the switch will close and then open, with the open delay time being proportional to the speed at which you are rotating the joystick. The slower you rotate, the more time the keyboard encoder has to detect the switch release to clearly distinguish that the next actual rotation is a new one, and not just a single extended single key press. If the joystick is spinning rapidly, the first keystroke is detected by the keyboard encoder as a switch press, but while rotating so fast, the switch is being re-triggered so quickly that by the time the keyboard encoder checks its inputs again, it still appears as if the original key press is being held (the keyboard encoder misses the on-to-off-to-on transition of the new rotation sent by the RI and

reads the new press somewhere in the middle, and since it missed the end of the previous rotation, it doesn't know if it is receiving the same initial key press, or a new one). Finally when the rotation stops, or slows down enough that the keyboard encoder can detect a release signal, the keyboard encoder can interpret individual key presses and send multiple signals to the computer instead of a single long key press.

The emulator doesn't mind if the rotary switch is held constantly, it will actually begin to act as an optical rotary joystick, or a spinner, with the character spinning as fast as the dial keys have been configured to rotate! This gives the player complete control over the character's motion based on how fast the joystick moves. The slower the joystick is rotated, the more control there is over what happens. If the joystick rotation keystrokes are observed in a text editor, the slower rotations should all register, and the faster the joystick is rotated, the less likely it is that the keystrokes will show up. This is just the way the editor chooses to handle the keyboard press and release signals. When spinning fast and having the switch appear to be pressed as if it were one long press, the editor may just see it as a single intended key press and ignore the other fast rotations, but the emulator handles it differently. When the emulator sees a constant key press with no detected release, it chooses to sustain the key press, thus sustaining the rotary action, and the character in the game continues to rotate while the joystick is rotating.

Circuit Description

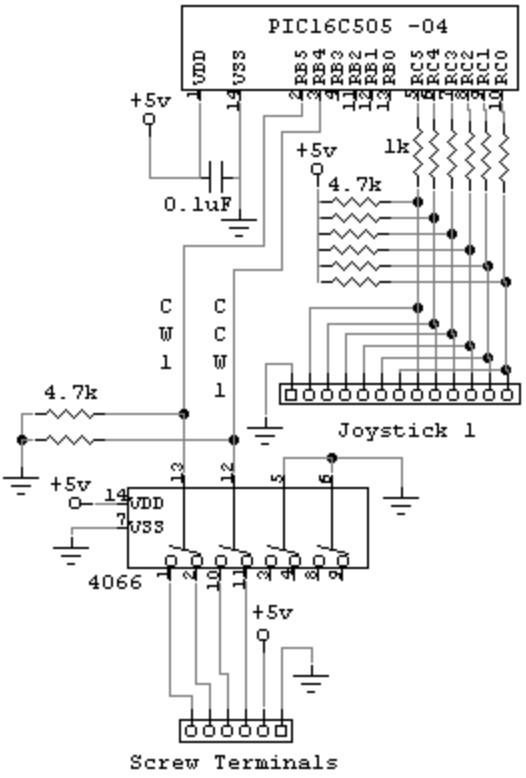
The Joystick And Connectors

The joysticks have a set of 13 pins at the bottom of their shaft, one for each of the 12 rotary switch positions, and a common connection for all 12 switches. The 13 pins on the joystick itself are in a different order than they are on the far end of the cable that plugs into the interface. The cable reorders the pins by relocating the ground connection, which is somewhere in the middle of the joystick's pins, to one edge of the connector at the other end of the cable. This causes some of the wires to shift towards the center of the cable by one position away from their original location on the joystick shaft.

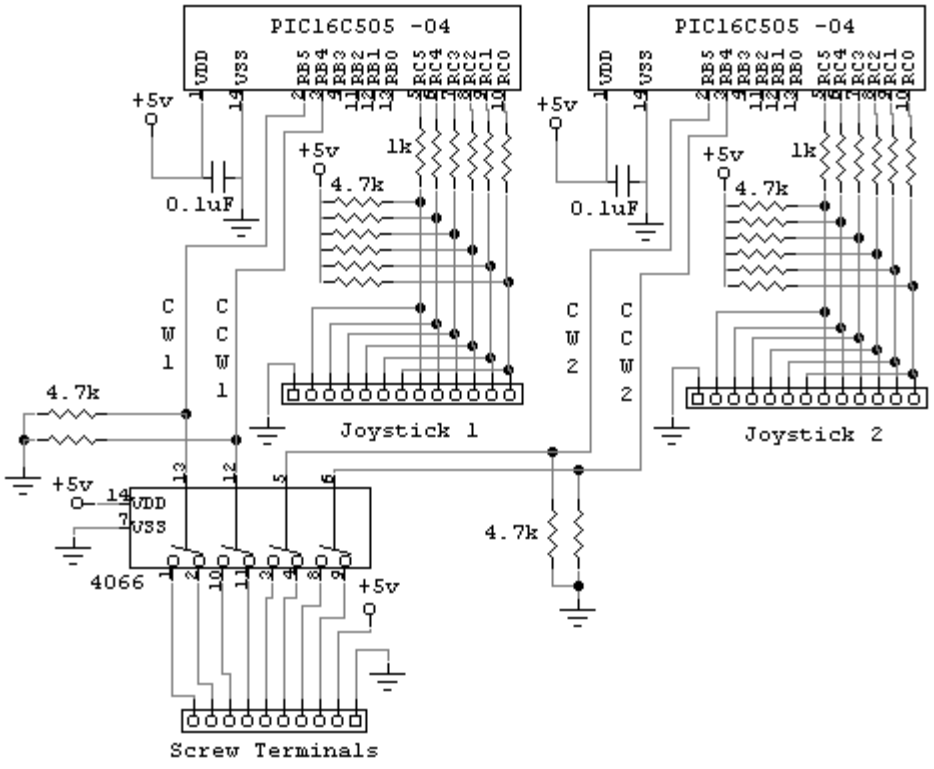
The result is that the end of the cable that plugs into the rotary interface board (or an original arcade board) has its pins all nicely arranged in a consecutive line, with the common connection at the end, and the other 12 pins in sequential order of rotation from an arbitrary starting point on the joystick. Each time the joystick rotates from one position to the next, the pin on the cable that was last active will become inactive, and the one directly beside it will be the new active one. When the joystick is rotated in the other direction, the pins will sequentially become active in the reverse direction along the cable.

When the cable is plugged into the joystick interface, the common connection on the cable is connected to the system ground. The pin that is currently "active" among the remaining 12 pins will result in that pin being shorted to the common ground. As the joystick rotates, the result seen by the interface is the same as having a row of 12 arcade switches connected to the rotary interface connector instead of having the joystick plugged in. With those 12 hypothetical switches, all of the switches would have one of their 2 contacts wired to the common ground pin of the interface's input connector. The other contact of each switch would be connected to each of the other 12 pins on the connector.

One switch would always be pressed down, shorting that connector pin to the common ground. When a change occurs, whichever switch was being pressed would be released, and then the switch right beside it, either to the left or right, would immediately be pressed and held down, shorting that new pin to ground. That is how the rotary joystick's 12 position switch operates, always having one switch constantly pressed and then changing to the next adjacent switch on either side as it is rotated. The rotary switch is responsible for routing one of 12 active pins on the connector to ground, to indicate its position to the micro controller (or original arcade motherboard).



Shown above is the complete single joystick circuit, and below is the modified unit with two-joystick support. Note that the joystick connector is shown in these diagrams as being wired with the grounded pin on the left and the first switch input (port C0) on the right. This is in reverse order from the physical wiring of the connector in a wired circuit. The important point is that the connections are wired in the proper sequence, but they are presented here in reverse order. (i.e. from the view of the topside of the circuit board with the angled connector pins pointing at you, the left most pin is wired to port C0 and the right most pin is grounded). If the pins are wired with the ground starting on the left, the joystick won't work. The actual order of connections on the screw terminals may vary between units and would be documented.



The Micro controller

The PIC16C505 chip monitors the activity on a joystick connector and determines which output screw terminal switches should be "pressed" as the joystick rotates. The "-04" in the part number of the schematic indicates the 4 MHz maximum clock version of the chip. Since micro controllers must have a definite high or low signal on their inputs, and since the inputs in this circuit are active low (because when the active rotational input switch is closed, the micro controller input is connected to ground), the default state on the inputs while the connector pins are not grounded must be a high (+5v) state. 4.7k pull up resistor networks are used to hold the input lines high until the rotary switch positions route an input pin to ground. There are also resistors directly in series with the input port pins to reduce the possibility of damage to the micro controller from static (ESD) or other interference. Any circuit that has data connections off the circuit board, such as inputs coming from a cable, should be equipped to handle all sorts of potential problems including ESD, noise, or shorts.

The output lines (ports B5 and B4) are activated to indicate a clockwise or counter clockwise rotation on joystick 1 or 2. They drive the 4066 IC, a quad analog switch providing four single pole single throw (SPST) switches that are electrically "pressed" via control inputs to open or close them. Each switch has two output contact pins on the IC, along with a third control input pin. When the control line of any switch is brought to a logic high, the result is that the two switch terminals assigned to that control line are closed, simulating a physical switch being pressed. When a logic low is present on the control line of the switch, the switch contacts are opened (the switch is released).

Since by default, the keyboard switches should be open (unpressed), the control lines of the 4066 switches must be kept low to open the switch contacts and break the circuit path through the switches. Pull-down resistors are used to tie these 4 control lines to ground, while the micro controller presents a high state when necessary, for a short while, to simulate pressing the switches connected to the keyboard encoder.

The only other component in the circuit is the 0.1uf capacitor across the micro controller's power supply pins. This capacitor should be located as physically close to the PIC as possible to help reduce noise from fast digital switching in the circuit, by providing a small local power reservoir for the chip. The capacitor is always charged through its connection to the 5 volt power supply, and when instabilities or power fluctuations occur on the supply, the chips will have a fresh 5 volt supply available from the capacitors for a very short time, long enough for the real power supply to stabilize. Also, due to the nature of capacitors, they act more like normal wires when any signal of a high frequency is passed through them. At low frequencies, the capacitor is less like a normal wire. When you apply this knowledge, a DC power supply is a zero frequency signal, it does not have a varying voltage level, just a solid 5 volts DC. The capacitor does not behave like a normal wire when it sees 5 volts DC, so putting a capacitor across the 5 volt supply does not short it out. But if you have noise also on the power lines, which is a high frequency signal, the actual noise treats the capacitor as a normal wire, thus the noise is shorted to ground on the power supply, or eliminated! This helps keep noise from entering the digital chips through the power supply wires.

Note that the micro controller does not use a crystal. This PIC has an internal RC oscillator running at 4 MHz. It is recommended to use an IC socket, especially for the micro controller, to minimize damage resulting from soldering, and to make any repair work easier.

Through the evolution of the design, it was determined that the best performance resulted from having two dedicated micro controllers instead of one single master to monitor multiple joysticks simultaneously. Every unit of time is critical in sampling the switches when they are rotating fast, and a missed sample could result in a misinterpretation of the data. Reverse rotations were a common problem previously because some high-speed rotations resulted in the chip missing some readings. By the time the chip made its next reading, the joystick had rotated so far that it had started to come full circle, which appeared simply as if it were rotating in the other direction when compared to the last registered sample. Suppose the joystick were at position 9 of 12 and

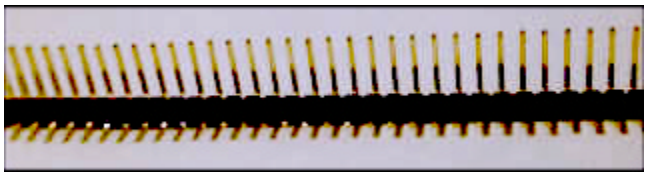
rotating fast in the clockwise direction. While the micro controller would be off reading the other joystick, the first joystick would proceed to position 10, 11, 12, and possibly on to looping back at 1 and 2. Finally the micro controller reads the first joystick and sees it is at position 2, which looks like it had gone counter-clockwise, compared to the previous reading of position 9. Several attempts were made at making an advanced calculation to determine the most likely 'real' direction of rotation, but in the end, the best way to get the accurate readings was to dedicate an entire chip to each process, allowing the circuit to keep up with the joysticks.

Although there are 12 switches on the rotary joystick, the connector on the RI board ties every sixth pin together, forming two groups of six consecutive pins. This is a clever design trick based on a suggestion by *Joel Simpson*. This method increases the efficiency of the circuit hardware and reduces the cost of the PIC micro controller. This new optimized hardware trick reduces the number of joystick input lines from the original 24, down to only 12 on the two-joystick interface. The original enhancement idea was to use a technique of grouping the inputs into four groups of three switches on each joystick. That would require only three input pins, but through much testing, it was determined that two groups of six inputs worked best.

The switches may be imagined to be laid out in a circle with 12 dots around it, and the only consideration for the circuit is in which direction the dots are being counted – not how many there are. By taking every 6th dot in the circle of 12 and tying them together, ending with two groups of six dots, the result is the same as having reduced the radius of the original 12 point circle down to one of only six dots around it. In such a smaller circle, it is still possible to tell which direction the dots are being counted, and it makes it much more efficient to design for such a simpler layout.

To analyze it visually, consider the 12 points of the bigger circle being numbered from 1 to 12. Originally, the design strategy was to monitor all 12 switches and watch which direction they were being activated in. By connecting the switches as shown in the schematic, assume you start at switch #1 at the end of the connector. Then activate the next switch, number 2. Then number 3, 4, 5, and 6. Now when you keep moving and activate #7, notice that it's tied to #1. The circle is repeating back at position #1 as far as the circuit is concerned, although the actual rotary switch is really only at position 7 of 12 and hasn't come full circle yet. When you move on to position #8, note it is tied to position #2. Position #9 is tied to position #3, and you have come half circle again in the 2 imaginary 6-point circles converted from the one bigger 12-point circle. The circuit turns one big circle into two smaller repeated circles, and the end result is the same.

The actual 13 pin connectors used on the interface for the joystick cables are standard 0.1" spaced IDC headers, similar to those used on disk drive connectors. Right angle pins were chosen so the cables would not plug straight down into the top of the board, but along the edge of the board from the side direction. The headers may be obtained in longer rows and broken off in 13 pin segments, or purchased in convenient lengths of 13. The shorter end of the pins is the end that connects to the board. The longer end of the pins, where the right-angle bend occurs, is the part that the joystick cable will plug into.

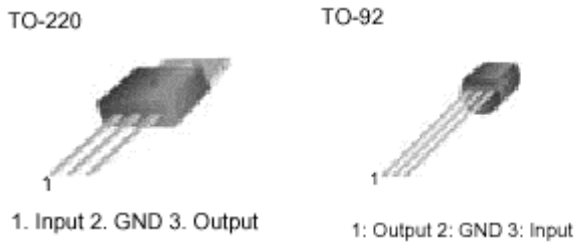


Optional Power Supply Circuit

The current drain of the circuit is only a few milliamps (maybe 5ma at peak draw), so this interface shouldn't be a burden on most power sources.

If a +5v power supply is not readily available from the computer, but a separate AC or DC adapter is available for use, it may be preferable to build an extra voltage regulator circuit to allow any adapter, AC or DC, to be used to safely power the interface. This adapter (or other voltage source such as a power transformer connected to the household power line) would have to be a few volts higher than the desired 5-volt output to compensate for voltage drops in the power supply circuit (diodes and the regulator itself). If the voltage regulator is not supplying an output close to +5 volts, the input power source may not have a high enough voltage. 7.5 or 8 volts should be enough.

The 7805 voltage regulator is shown below in two package formats. The TO-220 package is capable of supplying 1 amp, while the TO-92 version (78L05) can supply 100ma. Either version will work for the 5ma requirements of the interface.

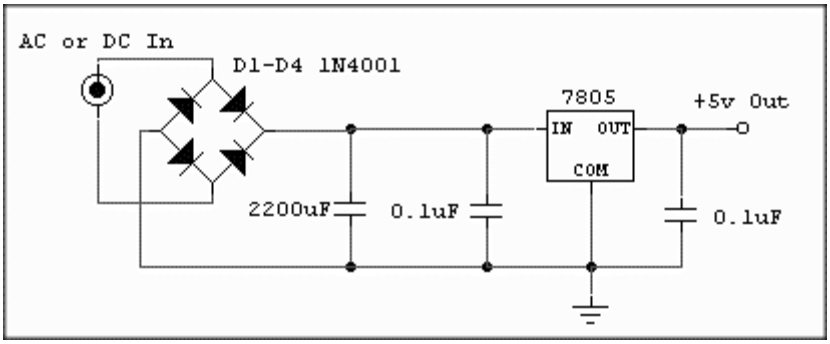


The 78(L)05 voltage regulator pin designations.

The maximum input voltage to the regulator before damage results may vary between device manufacturers. The data sheets for the Fairchild devices specify that up to 35 volts may be safely presented to the 1 amp TO-220 regulator, while 30 volts may be applied to the 100ma TO-92 version of the regulator. If a filter capacitor is connected across the bridge rectifier output, it must also be adequately rated to handle the voltage level that is expected from the adapter. Capacitors may be rated for use in applications ranging typically from 6.3 volts up to hundreds or thousands of volts. A 35 volt capacitor would be a good choice for a circuit powered from an adapter, assuming of course the adapter's voltage is nowhere near that high.

A heat sink is clipped or bolted to the regulator to enhance thermal transfer from the device to the surrounding air, preventing overheating. The simplest use of the device would involve connecting a voltage higher than 5 volts plus an extra 2 volts or so at the input and ground terminals, and then simply using the output and ground terminals as a regulated 5 volt supply. An extra capacitor close to 0.1uF is optionally connected across the output and ground terminals to improve voltage stability, while a capacitor connected across the input and ground terminals may be used if there is an appreciable distance between the rest of the power supply (adapter/diodes/filter capacitor) and the regulator IC.

Shown below is a typical power supply circuit allowing an AC or DC adapter to be used with the joystick interface. This circuit will maintain a steady 5-volt output regardless of the (steady) input adapter voltage used. The bridge rectifier, consisting of diodes D1 to D4 (1N4001 rectifier diodes) is included to convert AC to DC in case an AC adapter is used, and additionally provides polarity reversal protection for DC adapters.



A DC voltage may be connected with the + and – terminals on either of the input connector's terminals, always giving a proper DC voltage polarity at the output of the diode network. The + and – connections will be correctly routed to the output of the bridge rectifier due to the way the diode network operates.

How The Bridge Rectifier Works

The reason the bridge rectifier corrects the polarity of a reversed input connection is that only two diodes out of the four are ever in use at any given time. One pair of diodes will be conducting when the + and – inputs are connected in one way, while the other pair of diodes will operate if those inputs are reversed, and the original pair of diodes will then block current through themselves. Regardless of which pair of diodes are operating at any given time, the output will always have the + and – connections in the same order.

By observing the diode network, it can be seen that the point where either of the input adapter's connections meet the diodes, there is a junction of two diodes. One diode has its anode connected to that adapter connection, while the other diode has its cathode connected to that same adapter connector. Depending on whether that adapter input is + or -, one diode will block the current, while the other diode at that same terminal will allow current to pass through, directly to the output side of the diode network.

By analyzing the bridge rectifier with a theoretical example of having the adapter inputs ordered as +/- and then -/+, it could be verified that the output will always be the same +/- order. This is also how a bridge rectifier converts AC to DC. An AC voltage is one in which the voltage is constantly rising to a maximum (usually positive) voltage peak, then falling to a minimum (usually negative) peak, and then increasing back to the maximum peak. When the voltage level suddenly changes from a positive to a negative, the active pair of diodes will switch, where the conducting diodes will begin to block, and the blocking diodes will begin to conduct, and the proper + and – connections at the output are always properly ordered, thus the AC input with its switching polarity is converted to a single constant output polarity (DC).

A capacitor is connected across the output of the bridge rectifier to ensure that a smooth DC voltage is presented to the voltage regulator. The value required is negotiable depending on the characteristics of the adapter. A 1000uF may be adequate, or even no filter capacitor at all. The capacitor is only mostly necessary if the input voltage is AC. The resulting DC at the output of the bridge rectifier will not be steady in such a case. It will be rippling between some maximum and minimum voltage level, although the maximum and minimum happen to be both of the same (positive) polarity (hence DC). By including that filter capacitor, it will charge up to the highest voltage present in the rippling DC. When the rippling DC then begins to fall to its lowest voltage, the capacitor more slowly discharges towards that lower voltage, since it retains its charge instead of instantly dropping when the input adapter level fluctuates.

The 7805 regulator draws its input from this filter capacitor, thus being powered more towards the maximum voltage level of the DC ripple from the bridge rectifier when the input is AC. The capacitor may be thought of as a rechargeable battery that is powering the voltage regulator, but it drains very quickly. Every time the rippling voltage from the bridge rectifier reaches its most positive peak voltage, it is the same as fully recharging the battery, allowing it to be drained again by the voltage regulator. The capacitor is constantly

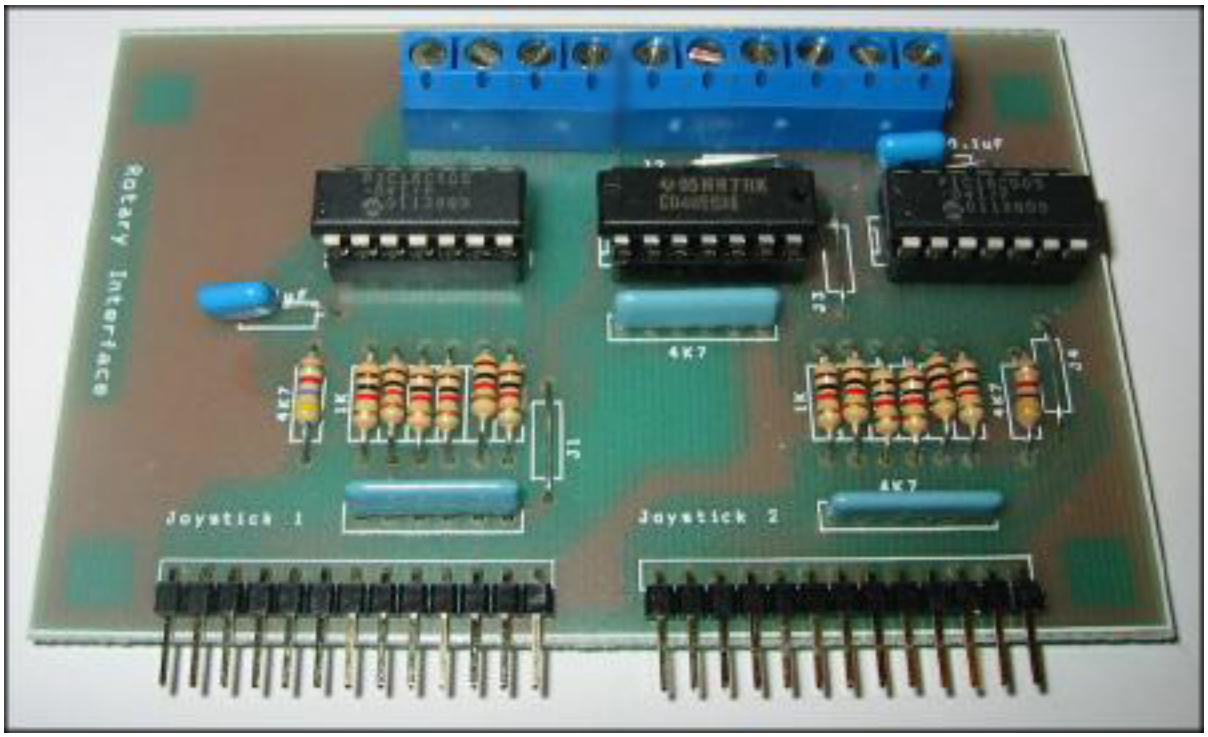
supplying the voltage regulator with a DC voltage, and is constantly having its stored charge refreshed by the highest peaks of the bridge rectifier voltage output.

If an AC voltage is not going to be used to power the interface, or if there is no concern about connecting a DC adapter backwards to the power supply, then the bridge rectifier and filter capacitor may be omitted, leaving just the voltage regulator and its 2 optional capacitors. The output voltage of the 7805 will be at safe levels for the interface's digital circuitry.

Appendix 1

Troubleshooting Guide

Troubleshooting



If the interface seems to not be responding during game play, it may be beneficial to load a text editor instead of using the game to observe the results of the interface's key press signals. During the designing process, it was much easier to watch characters appear in the notepad program to verify that the joystick was sending the desired key presses.

What is Wrong?

If the unit seems completely unresponsive, is there any power actually getting to the interface? Is it connected with the proper polarity? Is one joystick working ok but the other is doing weird and unpredictable things?

Generally, aside from a completely fried unit, the most likely problem is a crossed wire or solder bridge. Try manually moving any wires to be sure they are not touching neighbors. It is good to have a voltmeter and ohmmeter at times like this.

Any removable connections should be verified for secure contact. Try pushing on any socketed IC's to make sure they are firmly in place, make sure any removable test jumpers are firmly in place, and all screw terminals are tightly clamped.

Testing Interface Functionality Without A Joystick

To test the interface in a normal setup, but with no joystick plugged in, to eliminate the joystick or its cable from the faulty suspect list, the way to mimic a joystick is to connect a wire to the ground screw terminal and use it to probe the first 12 pins of the joystick connector. While a pin is being connected to ground, it is the same as having the joystick sitting in that rotational position. By moving the wire to the next pin in either direction, it is the same as having the joystick rotating one way or the other. The interface should respond while moving from any pin position to any other, not just directly adjacent pins.

Begin at the non-ground (left) end of the joystick connector on the interface. An output switch (screw terminal) may or may not be triggered when the first pin is connected to ground in this test mode. One reason that an output may not be triggered is that the pin being shorted may have been the last registered pin position, so by triggering that same pin again, the interface sees no rotational change and doesn't set an output switch. It is the same as if the joystick had been sitting stationary all the while. If there is no output triggered when you first touch a pin, there should be an output triggered when you touch another pin except the ground pin.

As you move the ground wire along the 12 pins in one direction, a constant output switch should be activated each time. When the wire is moved in the other direction along the 12 pins, another output should be triggered each time. It is expected to see some false triggering by manually touching a wire and accidentally hitting the neighboring pins, but overall there should be one switch triggered repeatedly in a constant direction along the pins.

If there is output activity but it is other than expected, such as having one output triggered while grounding a pin as expected, but then seeing another output triggered upon releasing that same pin (and not touching any others), there may be a short along the wires between the connector and the micro controller inputs and pull up resistors. This would force that faulty pin to a logic state other than desired, making it seem that false rotations have occurred.

Testing Input Connectors

With power applied to the interface, connect a volt meter with one probe to ground and use the other probe to check the voltage at each of the first 12 connector pins with no joystick cable plugged in. The 13th pin is ground so there should be no voltage. All other pins should read close to +5 volts since the 4.7k resistors are pulling them up. If any of the first 12 pins on each joystick connector read closer to 0 volts than 5, there may be a short or a faulty pull-up resistor.

To verify, it may be helpful to use an ohmmeter to check the pull up resistors. Connect one probe to the +5v terminal (with no power applied) and use the other probe to test each of the first 12 pins on both joystick connectors. They should all read close to 4.7k. One side of each resistor is connected to one of the 12 connector pins, while the other sides of all resistors are connected to the +5v bus. Resistance measurements may only be made in a circuit that is **not powered**. The ohmmeter's battery provides the required power to make the measurements.

Testing Outputs

To verify that the micro controller is sending the proper output signals, test the outputs directly from the micro controller port B5 and B4 pins (pins 2 and 3) where they run to the 4066 IC and make sure there is activity during rotations. An LED with a resistor (around 330 ohms) to ground may give a better indication than a voltage/current meter for the short duration that the outputs will be active. Make sure the LEDs are connected with proper diode polarity (connect directly to +5v and ground first, using the resistor always) or they will always be off if they are backwards during the test. If the outputs seem to stay low when they should be high, verify that there are no shorts around the output lines and their pull down resistors, as well as the input control lines to the 4066 chip. If the outputs of the micro controller seem to be giving proper outputs but the output switches (screw terminals) are not responding, verify that the micro controller outputs are reaching the 4066 input pins (13, 12, 5, 6).

Try simply making sure also that the keyboard encoder itself is working. Manually short the two screw terminals of each output switch and see that the keyboard encoder sends the proper keystroke.

Appendix 2

Parts List

Parts List – 2 Player Interface

Description	Qty	Digikey Part #
PIC 16c505 Micro Controller 4MHz OTP (unprogrammed)	2	PIC16C505-04/P-ND
4066 Quad Bilateral Switch IC	1	296-2061-5-ND
1 KΩ ¼ Watt Resistor	12	1.0KQBK-ND
4.7 KΩ ¼ Watt Resistor	2	4.7KQBK-ND
4.7 KΩ SIP Resistor Network (5 bussed resistors, 6 pin)	3	773061472-ND
0.1µF Capacitor	2or3	P4525-ND or similar
13 Position Header, 0.1" Spaced	2	A19401-ND
14 Pin IC Socket	3	ED3114-ND
2 Position Screw Terminal Block	5	ED1601-ND

Parts List – 1 Player Interface

Description	Qty	Digikey Part #
PIC 16c505 Micro Controller 4MHz OTP (unprogrammed)	1	PIC16C505-04/P-ND
4066 Quad Bilateral Switch IC	1	296-2061-5-ND
1 KΩ ¼ Watt Resistor	6	1.0KQBK-ND
4.7 KΩ ¼ Watt Resistor	1	4.7KQBK-ND
4.7 KΩ SIP Resistor Network (5 bussed resistors, 6 pin)	2	773061472-ND
0.1µF Capacitor	1or2	P4525-ND or similar
13 Position Header, 0.1" Spaced	1	A19401-ND
14 Pin IC Socket	2	ED3114-ND
2 Position Screw Terminal Block	3	ED1601-ND

Misc Items:
PC Board
Hookup Wire 22 AWG Solid
Solder & Iron
Wire Strippers/Cutters
PIC16C505 Programmer & Source Files if required

Notes:

- The schematic shows several 4.7KΩ resistors – six on joystick connector 1, six on joystick connector 2, and four on the 4066 IC inputs. The parts list shows two 4.7KΩ resistors, and three 4.7KΩ SIP resistor networks (5 resistors in each).
- For the six 4.7KΩ resistors on each joystick connector, one 5 resistor network is used, in addition to a single 4.7KΩ resistor to make up the sixth resistor for the connector.
- For the four 4.7KΩ resistors on the 4066 switch inputs, a single 5 resistor network is used, with one of the 5 resistors left unused in the network.
- If the PIC 16C505 chips are not purchased from me, they will require programming
- There is an extra 0.1µF capacitor listed in the parts list, this is an optional bypass capacitor that may be added across the power pins of the 4066 IC if desired. It hasn't proven necessary but if the hardware is subjected to excessive noise, resulting in strange behaviour, it may be recommended to add a capacitor across the power pins of the IC.

Appendix 3

Reference Material & Contact List

Links
(Current as of February 2002)

Support Email (questions or comments): hybrid_x@yahoo.com

Product Page: http://connect.to/rotary (global short URL)
http://members.rogers.com/druins22/ls30 (main long URL)

Semiconductor Data sheets

PIC 16C505 Micro controller (download PDF document)
http://www.microchip.com/1010/pline/picmicro/families/16c5x/devices/16c505/index.htm

4066 Quad Analog Switch IC (download PDF document)
http://www.fairchildsemi.com/pf/CD/CD4066BC.html

78(L)05 Voltage Regulator (+5 volt) (download PDF document)
http://www.fairchildsemi.com/pf/KA/KA7805.html (1 amp TO-220)
http://www.fairchildsemi.com/pf/KA/KA78L05A.html (100 milliamp TO-92)