Mini-Kit Assembly Manual and PS/2 Theory of Operation

Introduction to the Mini-Kit

Thank you for purchasing the KibaCorp Min-Kit interface. KibaCorp is dedicated to Microcontroller education for the student, hobbyist and practicing engineer. With this interface you will be able to use a Rotary Encoder and PS/2 Keyboard with your 16 Bit Micro Experimenter (Experimenter for short) as an integral part of your future 16 bit experiments.

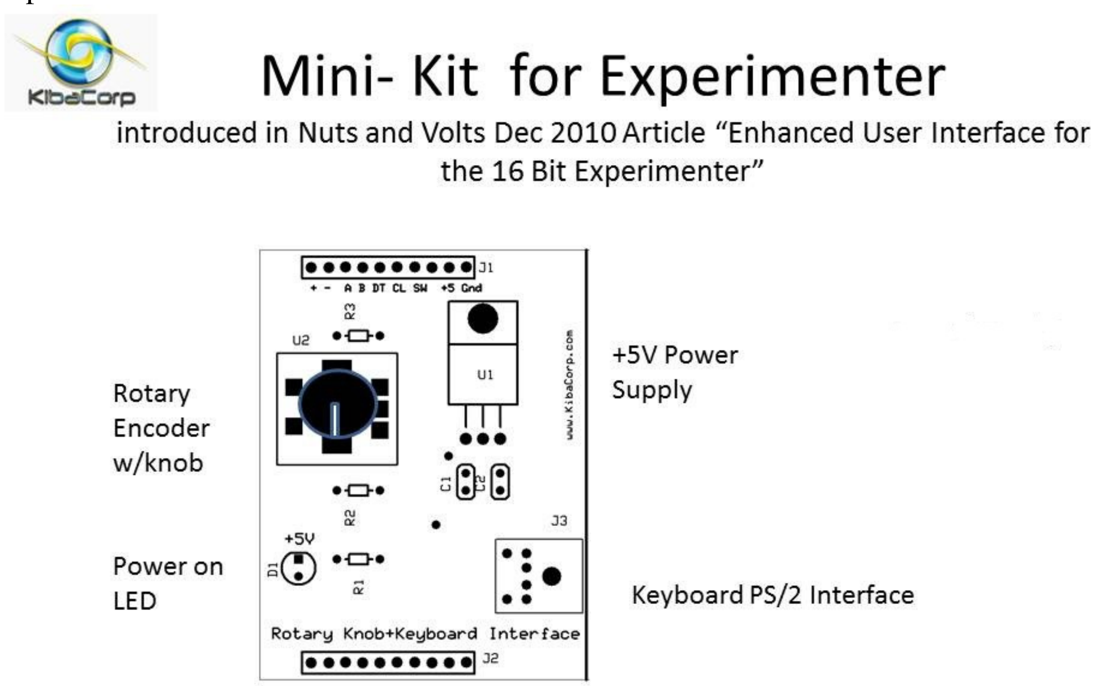


Figure 1 Mini-Kit

Mini-Kit parts List

A picture of the kit and a list of all components are given
They are as follows:

- (1) 1K 1/8 watt resistor R1

- (3) 10 K 1/8 watt resistors R2, R3, R4
- (1) LM78M05 +5 V DC 500 ma linear regulator U1
- (1) Rotary Encoder with built in switch U2
- (2) 10 pin .1" Headers J1 and J2
- (1) Rotary Encoder rubber knob
- (1) Green LED D1
- (1) 10uf capacitor C1
- (1) .1uf capacitor C2
- (1) 6 pin DIN PS/2 Female Connector J3
- (1) Printed Circuit Board

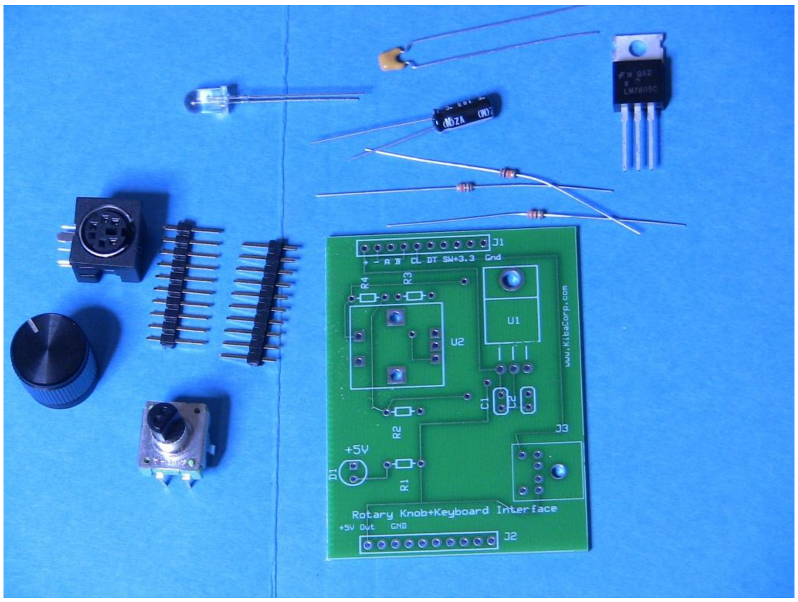


Figure 2 Mini-kit Parts

The Rotary Encoder theory of operation and demos were covered in Nuts and Volts Dec 2010. Included in this manual (Appendix A) is the theory of operation covering the PS-2 interface theory of operation. All example software demos are included in the CD-ROM. The whole 16 bit Micro Experimenter system in operation with the Mini-Kit is shown in the figure below.

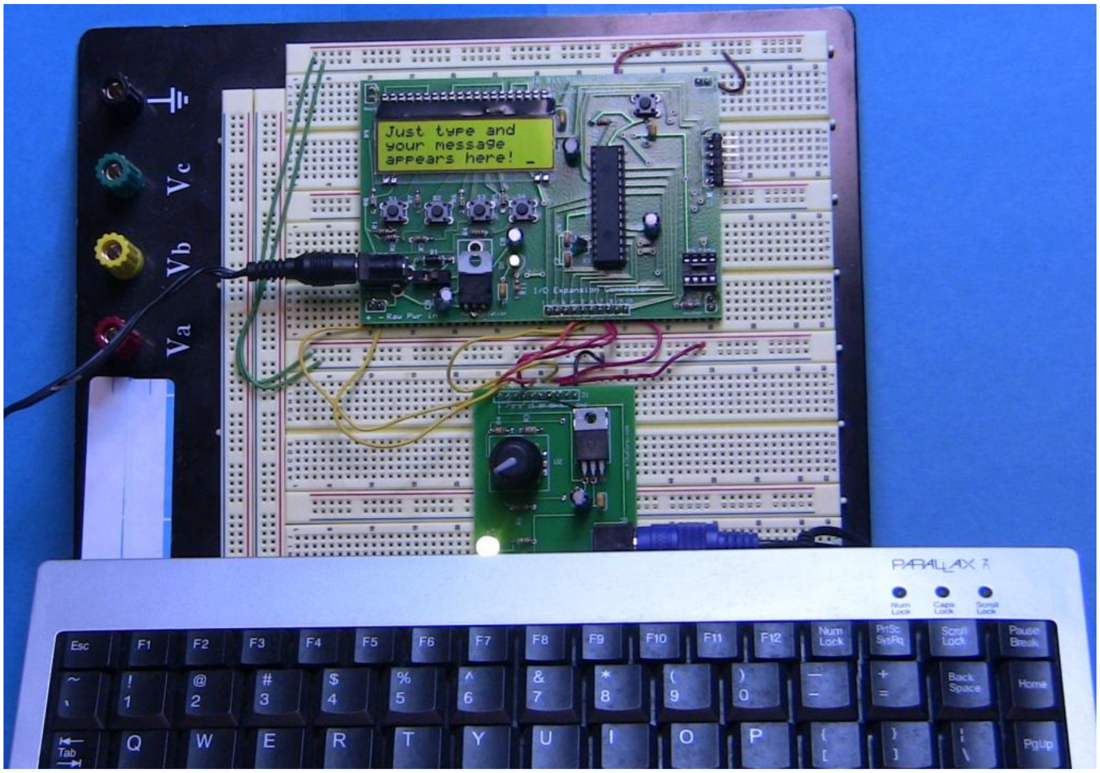


Figure 3 Mini-Kit in operation

Mini-Kit Theory of Operation

The mini-kit contains a Rotary Encoder Knob and interface, a PS/2 Keyboard interface and a +5V DC regulator. The +5V regulator is needed to provide the required voltage levels for the PS/2 Interface (to power up the keyboard and provide proper voltage levels for the PS/2 interface). The +5V output is also available to the user to power up any external +5V hardware during for prototyping. The Experimenter itself is a +3.3V device but has +5V tolerant lines that are available on its I/O Expansion Bus for the Mini-kit connections. Recommended hook-up lines in the Mini-Kit hook up diagram (see Figure 4) are provided, but in general, any Experimenter pin that does not include an analog capability can be used as well. A full description of the Experimenter I/O Bus pin capability is given in the 16 bit Micro Experimenter Assembly Manual that was on the CD-ROM included with the Experimenter Kit. All necessary manuals are offered as free downloads from the KibaCorp web site www.kibacorp.com
The Mini-Kit interfaces J1 and J2 are detailed as follows:

- J1 Pins
    1. \+ raw voltage input into min-kit +6V DC or higher
    2. \- raw voltage ground in
    3. A A output from Rotary Encoder (+3.3V to 0 V)
    4. B B output from Rotary Encoder (+3.3V to 0V)
    5. CL Clock output from keyboard (+5V to 0V)
    6. DAT Data output from keyboard(+5V to 0V)
    7. SW Rotary knob push switch
    8. +3.3V Regulated +3.3V in

9. Not used
10. Ground

- J2 Pins
  1. +5V output from regulator
  2. Not used
  3. Ground
  4 Not used
  5 Not used
  6 Not used
  7 Not used
  8 Not used
  9 Not used
  10 Not used

Assembling the Mini-Kit

The tools necessary are a soldering iron, dykes, and solder.

1. Layout all the components from kit bag on your work area. Place the PCB board silk screen labeling so the top side silkscreen is visible as shown in Figure 5 below

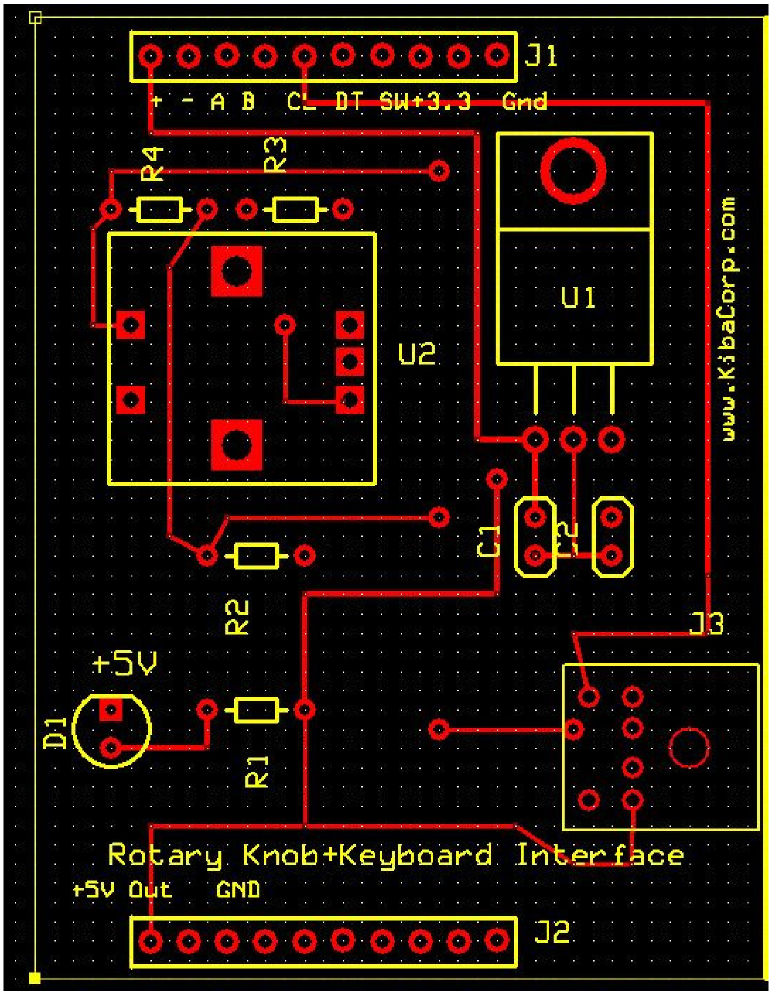


Figure 4 PCB Silk Screen

2. The first items to solder are J1 and J2 .1" 10 pin headers. Insert headers into their positions on the board (again examine silkscreen).These headers are inserted through the bottom side of the board and soldered from the top layer.
3. Place regulator into U1 position as shown on top side silkscreen of PCB. Solder from bottom of board. Use dykes to cut excess leads.
4. Place rotary encoder into U2 position as shown on top side silkscreen of PCB. Solder from bottom of board. Use dykes to cut excess leads.
5. Place keyboard PS/2 connector encoder into U3 position as shown on top side silkscreen of PCB. Solder from bottom of board.
6. Place 1K resistor (Brown, Black, Red bands marking) into R1 position as shown on top side silkscreen of PCB. Solder from bottom of board. Use dykes to cut excess leads
7. Put LED into D1position position as shown on top side silkscreen of PCB. Solder from bottom of board. Use dykes to cut excess leads. Note that D1 is polarized with flat side (cathode) marked on silkscreen.
8. Put 10uF capacitor into C1 position as shown on top side silkscreen of PCB. Solder from bottom of board. Use dykes to cut excess leads. Note that C1 is polarized with positive side (+) marked on top side silkscreen.

9. Put .1uF capacitor (Marked 104) as shown on top side silkscreen of PCB. Use dykes to cut excess leads
10. Time to check +5V output. Apply +6V to up to +9V to + pin1 of J1 with ground to pin 2. Check that Green LED is turn on when power is applied. Use a voltmeter to check pins 1 and 3 of J2 for +5V. If +5v is not measured or the LED is not lit double check C1 placement and all solder joints. Reheat solder joints if a cold solder joint is detected. Next again check that LED is turn on when power is applied. If LED does not light check for its correct orientation on board relative to cathode top level silkscreen marking.
11. Next solder in the three 10K resistors (Brown, Black, Orange marking bands) into their respective positions on top of board (see silkscreen). Use dykes to cut excessive leads.
12. Finally mount knob onto encoder shaft.
13. The mini-kit has completed assembly. It should appears as follows:

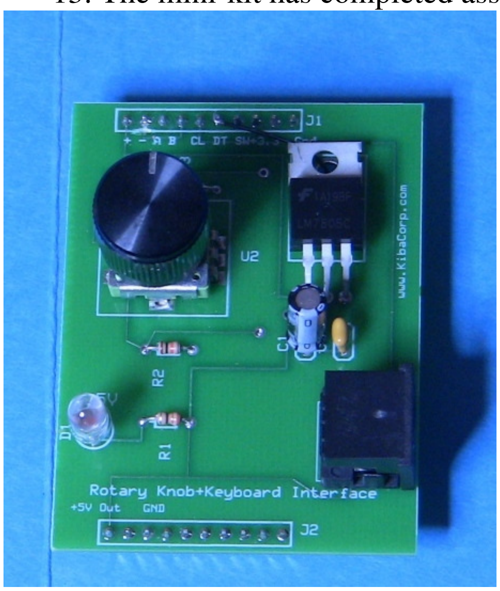


Figure 5 Mini-Kit Assembled

14. Wire mini-kit to Experimenter using provided hook-up diagram. Finally program your Experimenter with demo code provided and verify operation. On CD-ROM Rotary Demo and Scroll Demo to verify Rotary Knob operation, and Keyboard Demo to verify keyboard operation. You will need to purchase your own PS/2 keyboard to run the keyboard operation.

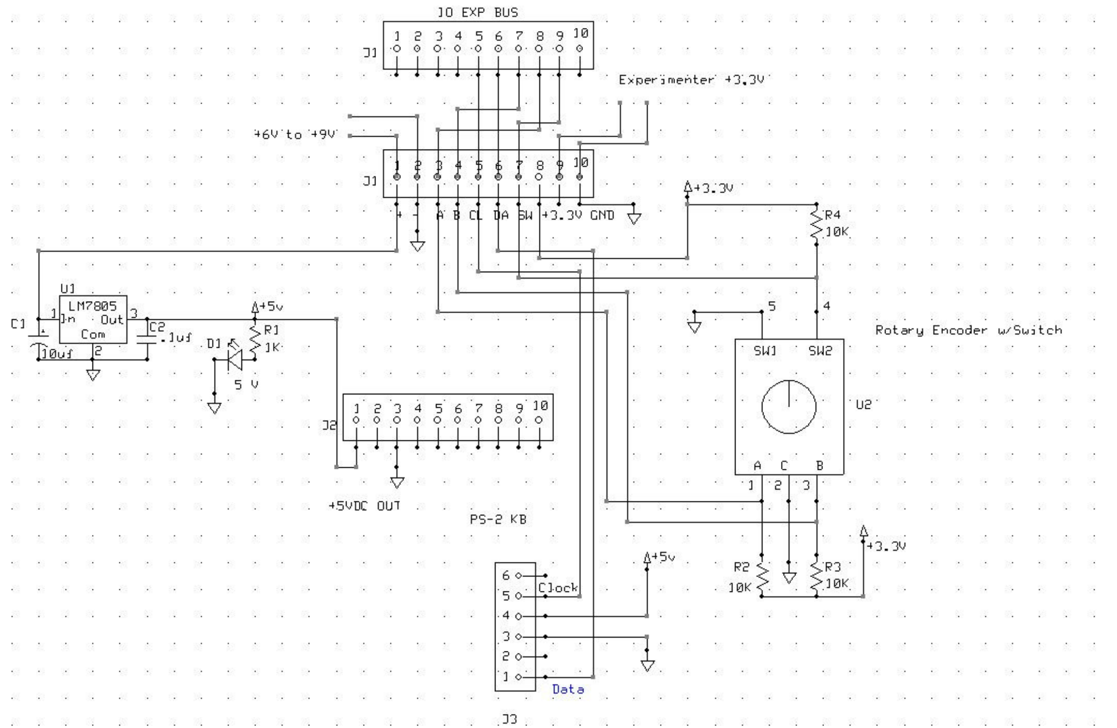


Figure 6 Hook up Diagram

Appendix A -Using Mini-Kit to Interface with a PS/2 Keyboard

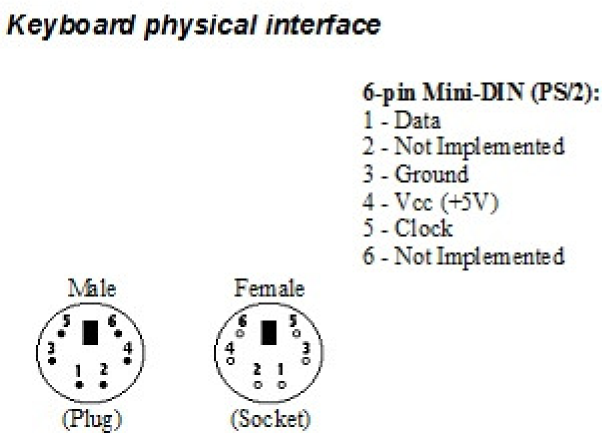
Introduction

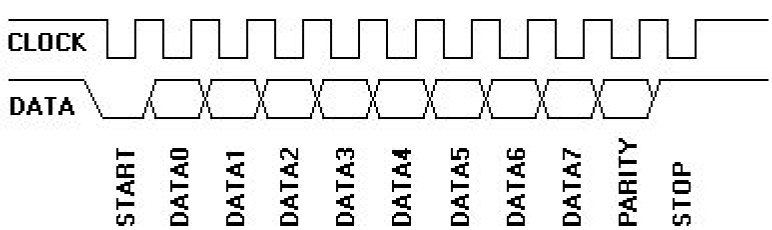
The keyboard is a powerful input capability for Experimenter Applications. With the advent of USB keyboards the PS/2 versions have pretty much been replaced, but still exist in large quantity and are cheaply available. Here we will discuss PS/2 operation theory and integrating a PS/2 keyboard to our Experimenter. Upon completion of this integration you should be able to type on the keyboard and the keyboard characters will appear on the Experimenter LCD.

Between the PS/2 and the Experimenter there is an extensive host-device protocol involved in communications; however in our case we will simply stick to the basics of receiving key scan data from the keyboard, deciphering these key scan into ASCII and then displaying this data on the Experimenter LCD Display. We will avoid any of the possible 17 host-to-keyboard command features.

PS/2 Physical Interface

Let's look at the physical interface, the keyboard requires +5V to operate, clock is generated by the keyboard and data from keyboard or host. The communication uses a bidirectional synchronous serial protocol. Because the Data line is bi-directional, data can originate on either side of the interface. Think of the Data Line as a bidirectional "bus" between host and keyboard. Again to keep things simple, we will only be involved with data received from the keyboard. The bus is "idle" when both clock and data lines are high (open-collector). This is the only state where the keyboard is allowed to begin transmitting data. The host has ultimate control over the bus and may inhibit communication at any time by pulling the Clock line low.





All data is transmitted "framed" one data byte at a time and each data byte is sent in a frame consisting of 11-12 bits. These bits are:

- 1 start bit. This is always 0.
- 8 data bits, least significant bit first.
- 1 parity bit (odd parity).
- 1 stop bit. This is always 1.
- 1 acknowledge bit (host-to-device communication only)

The parity bit is set if there is an even number of 1's in the data bits and reset (0) if there is an odd number of 1's in the data bits. This is used for error detection. Data changes during the clock high state and it valid when the clock line is low. The baud rate is not standard and can change from unit to unit; typical clock frequency can range from 10 - 16.7 kHz.

Integrating the PS/2

Ok there is bunch of things we need to do here to retrieve a key pressed on the PS/2 and to display this key as a character on the Experimenter LCD. First we need to recover key from the keyboard when it transmits. The key is in the form of a data frame (see interface description earlier of 11 bits). To do this we need to sample the PS/2 data line and clock line simultaneously and then determine when the PS/2 clock falling edge is detected (this is where the keyboard outputs valid data). We use the Experimenter PIC24F Microcontroller timer 4 in an interrupt mode to generate the sample time .The PS/2 clock can be approximated as a square wave with 50% duty cycle and a period of about 62.5 to 50 microseconds. If the Timer 4 interrupt is set to every 25 microseconds we are guaranteed during a timer 4 interrupt service routine to get at least one sample of PS/2 clock between falling and rising clock edges (good enough to capture an output PS/2 data sample).

This is not all that Timer 4 interrupt service needs to do. Once a valid data sample is taken it needs to be processed as a valid bit in a frame. The frame (11 bits -as shown in the physical interface description) needs to be validated for correct start, stop, and parity before being passed on as a legitimate key scan code. Key scan codes are those raw scan reports representing row and column of detected key actions generated by the keyboard when scanning its matrix of 80-101 keys.
The Timer4 interrupt service is still is not done. Once a valid key code has been received then the service needs to buffer up the key code and then alert a non-interrupt application process that a valid key code has been received and a key code is waiting within the buffer for the non-interrupt application to process. The non-interrupt application has the responsibility to prepare the key code for display by looking up its associated ASCII representation before issuing it to the LCD display. The whole process is captured as a flow chart in the attached figure. Demo software is provided for the Experimenter (see KEYBD.MCP). The required hook-up is shown in Figure 6.

High level Keyboard /Timer Experiment Flow chart