ibm-capsense-usb

Installation: Model-F-USB-Rev2

Tom Wong-Cornall
tom@wongcornall.com

July 9, 2014

Contents

1 Compatibility

The Model-F-USB-Rev2 is, as far as has been determined so far, compatible with all IBM Model F keyboards, with the exception of XT keyboards and other keyboards that have integrated controllers. 3178s and derivatives (such as 3104s etc.) are compatible, with the addition of an adaptor and some modifications; see Appendix A on page 17.

In general, as long as the keyboard has a 3.96mm-pitch 30-position ribbon cable, and conforms to the pinout in Figure 1 on the next page, then it should work. Keyboards that have less than the 16 columns and/or 8 rows are still compatible; the missing keys can just be ignored.

2 Required parts and tools

In addition to the Model-F-USB-Rev2 controller, you will need a micro-USB cable.

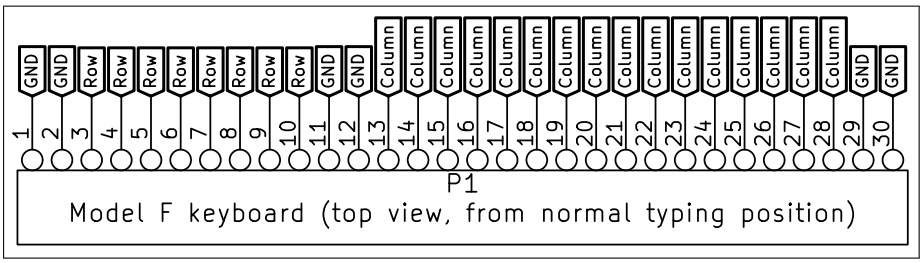

Figure 1: Pinout of Model F ribbon cable

A medium flat-head screwdriver and a small adjustable wrench (or tiny socket set) will be necessary to open the case and remove the controller.

As the Model Fs have a controller soldered to a ribbon cable, a soldering iron and some soldering equipment will be needed to remove the old controller and install the new one. At a minimum, a solder-sucker will do the job of desoldering the old controller; the addition of good quality solder wick will make the job much easier and help prevent damage to the old controller and the ribbon cable.

In addition, the USB software utility ("IBM Capsense USB Util") will be required to assign keycodes to each key (see Section 5.2 on page 6).

3 Precautions

Both the keyboards and the new controller are reasonably robust, but some common-sense must be used. Be sure to observe the following points.

3.1 Static electricity

The controller is a bare circuit board, so can be damaged by static electricity. Try to keep your fingers off the components and pins, and discharge any static first by touching something big and metallic, like the top-plate of your keyboard. Be extra-cautious in low-humidity conditions, as you can build up a couple of thousand volts in static electricity simply by walking across a room.

The same point goes for the original IBM controller; it would be a shame to damage something that is now irreplaceable.

3.2 Removing/reinstalling parts

Don't force anything. Nothing on these keyboards should be so stubborn that you need to use any strength to dismantle it. Check that you have removed all screws and wires/cables (including carefully desoldering the ribbon cable) before removing the original controller.

Be careful of cross-threading screws when they are re-installed, and make sure all washers are retained.

3.3 Cabling

Don't kink or fold the micro-USB cable. If possible, try to make use of the existing strain relief (for the original cable) to ensure tugging on the USB cable after installation won't pull the cable out of the socket, or rip the socket off the controller.

Don't bend the 30-core ribbon cable too much, especially near where it is soldered into either the pad card or the controller. It is somewhat brittle, and it is possible to snap the cores if it is treated roughly.

3.4 Short circuits

Ensure the controller won't come into contact with any metal parts or wires that could cause shorts.

After the new controller is installed, carefully inspect the ribbon cable, taking care to repair any damage to the insulation with electrical tape or similar.

Inspect both sides of the new controller, making sure there are no solder bridges between pins or to components on the controller. Mop up excess solder with solder wick.

3.5 Cleanliness

Capacitive switching is very sensitive and deals with tiny signal levels. Dirt or other contamination can intefere with the sensing and operation, especially if it reaches the sense pads underneath each keyswitch on the pad card. Make sure the keyboard and controller are both clean and tidy.

4 Removal of old controller and preparation

4.1 Opening case

Each keyboard is slightly different, but in general the bottom half of the case can be removed with four or so screws on the underside of the keyboard. With most keyboards the top half of the case can be left in place.

4.2 Removing old controller

First remove the external cables. The main communication cable will attach to the original controller via a push-in connector to some right-angle pin headers.

Some keyboards—probably just the PC-AT—have lock LEDs, connecting to the same right-angle pin headers. Unplug these as well; in Section 9 on page 12 you will see how to connect them via the expansion header.

Now you have the tricky job of desoldering the ribbon cable. There are two nice articles on desoldering on http://pinballrehab.com. Gentle pulling of the ribbon cable while heating the solder joint may help, but be careful not to pull too hard and damage the insulation, which will be very soft.

In the worst cases, it may be easier to simply cut the ribbon cable close to the old controller; you should be warned that stripping the insulation cleanly from the resulting cable is harder than it might seem.

Try to keep the wires unfrayed and straight—remember you will have to push them through the holes in the new controller.

Once the ribbon cable is desoldered and clear of the old controller, remove the two screws holding it onto the brackets. Repair any damage to the ribbon cable with electrical tape or similar so that it won't cause shorts on the new controller.

The old cable can be now be removed completely from the keyboard. When removing the old cable, be careful to retain the strain relief, as it may come in useful for retaining the micro-USB cable.

5 New controller installation

5.1 Mechanical installation

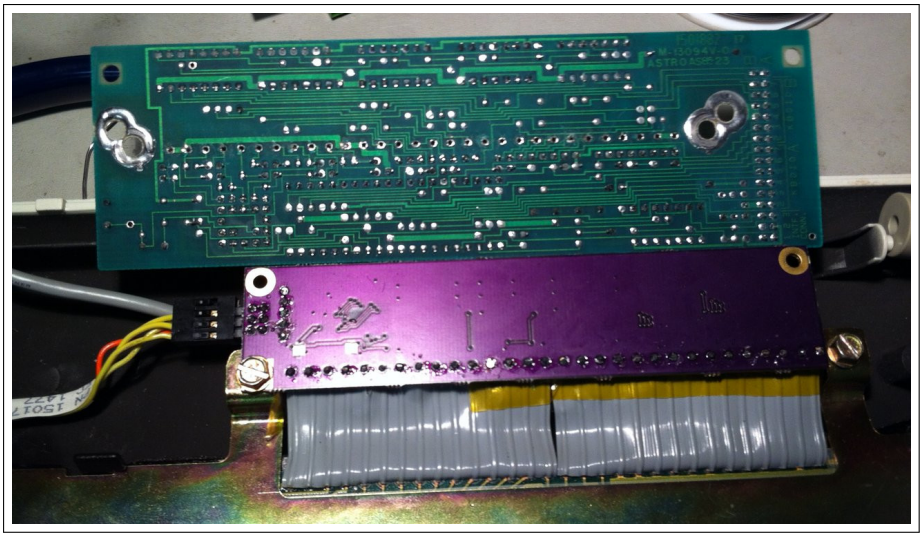

Figure 2: New controller (Rev1) correctly installed into PC-AT, next to old controller

1. Route the micro-USB cable. If possible, make use of the original strain relief. Leave enough slack so the cable will almost meet the controller bracket on the right-hand-side of the keyboard (when viewed from the top).
2. Correctly orientate the controller. The text/logo on the controller should be visible and readable if you were viewing the keyboard from a normal typing position (if the top cover is still in place, imagine you are looking through it). See Figure 2 for an example.
3. Insert the ribbon cable through the top of the controller and out the back—the same direction the ribbon cable passed through the old controller
4. Carefully solder the ribbon cable in place. Be careful not to cause any bridges/shorts on either side of the controller.

5. Screw the new controller to the brackets. You will probably only be able to get one screw in, on the right-hand-side of the controller (when viewed from the top of the keyboard, which should be plenty; on most keyboards this will be the lower screwhole, but on 6019284 keyboads (4704 "Kishsavers") you will use the top screwhole.

6. Make a final check to see all cores in the ribbon cable have good solder joints, without bridging (check some of the small surface-mount components on the other side), and that the screw is making good contact (good grounding to the keyboard top/bottom plates is very important)

7. With the PC-end disconnected, plug in the USB-micro cable, and check that your OS recognises the new USB device

5.2 Software installation

The PC-side software can be obtained (along with firmware updates, schematics etc.) from http://downloads.cornall.co/ibm-capsense-usb.

In general, you want the latest version. Check, however, that your version matches at least the `MINOR` version number (version numbering being `MAJOR.MINOR.PATCHLEVEL`) of the firmware (you will be able to see this once you have started the software).

Running the software should detect the board immediately, and begin reading scancodes and settings back from the keyboard, before presenting the user interface. If multiple ibm-capsense-usb controllers are plugged in, a menu will be presented to allow selection between them.

5.2.1 Linux and BSD

Linux and BSD users should compile their own binary from the source distribution (files named `ibm-capsense-usb_x.y.z.tar.gz`).

Prerequisites include Qt (preferably version ≥5.1.0, although it can be compiled with some limited functionality on Qt 4.8) and hidapi. These are probably available from your package manager.

Enter the `ibm-capsense-usb_x.y.z/src/util/` directory, then it should be a matter of simply typing `qmake` the first time to generate a makefile, then `make`. The resulting binary is named `ibm_capsense_usb_util` and placed in `ibm-capsense-usb_x.y.z/src/util/src`.

5.2.2 Windows

Pre-compiled binaries are available for most versions. The files are named in the format `ibm-capsense-usb-util_x.y.z.zip`.

Plugging in the keyboard the first time may cause some versions of Windows to spend a long time searching for "drivers". Experimentation seems to suggest clicking "Skip" is perfectly safe.

There is no installer or setup necessary; running the file named `ibm_capsense_usb_util.exe` should suffice.

5.2.3 Mac OS X

Pre-compiled binaries are available for most versions. The files are named in the format `ibm-capsense-usb-util_x.y.z.dmg`. These can be installed as normal disk images.

6 Controller setup

All setup is performed using the software utility.

When you first start the utility, you will be presented with the screen shown in Figure 3.

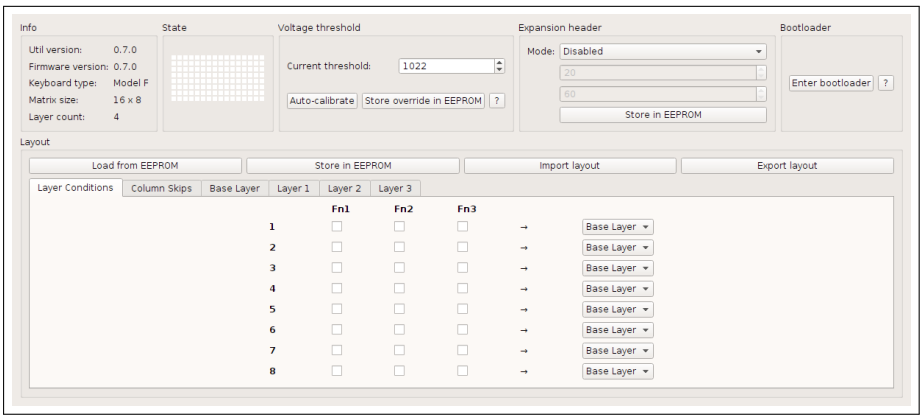

Figure 3: First startup of GUI util

First, check the keyboard type and versions are as expected (in the "Info" section of the util).

6.1 Initial voltage threshold setting

6.1.1 "Current threshold"

Check the "Current threshold" value under "Voltage threshold". This should be sitting at 1022. Using the "Auto-calibrate" button will not do anything useful at this stage—don't try it just yet.

6.1.2 "State" grid

Next, take note of the "State" section. This grid of 128 cells—16 wide and 8 high—represent the sensed state of each node in the keyboard matrix. With a voltage threshold of 1022, every node should be white (meaning *released*). *Pressed* nodes are shown as dark grey cells.

Mousing over a particular cell will show its co-ordinates within the matrix.

Pressing a key will elicit no change in this "State" overview. You must now set the voltage threshold so keys that you press will correctly show up in the grid

6.1.3 Setting threshold

Gradually lower the threshold towards 0, stopping to press keys now and again to see if they register. The permissible range is 0–1023, but you will probably find the keys will correctly register somewhere within a wide band between 100 and 150.

At some point, you will notice some keys will appear as pressed that don't correspond to physical keys. There are usually no more than 3 or 4 of these. These will come into play later on with auto-calibration. Figure 4 shows a keyboard with a calibration node registering at coordinates (11, 8). If your keys are registering correctly but you don't see a pressed "calibration key", keep lowering the threshold—you will not be able to autocalibrate without finding one.

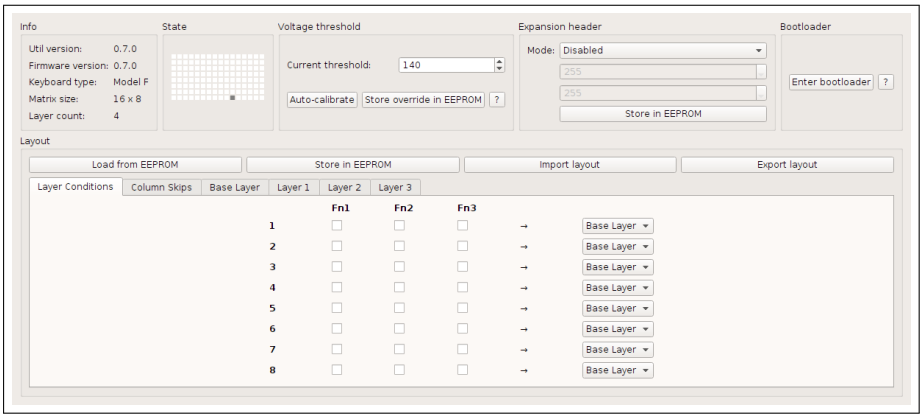

Figure 4: PC-AT keyboard with sense threshold correctly set

Try to find a threshold where pressing keys across the keyboard register correctly, don't falsely cause their neighbours to trigger (try pressing many keys at once in different combinations—full NKRO should be achievable), and don't flicker on/off. For now, remember the value in case auto-calibration fails later on.

6.2 Assigning base scancodes

Neither the controller nor the software know anything about the physical layout of your keyboard just yet. You must assign scancodes to each key before it will be recognised in your OS. Assigning certain special scancodes is also necessary for auto-calibration to work when you plug your keyboard in, or reboot your computer.

6.2.1 Nodes with keys

Change to the "Base Layer" tab. Each node is represented by a drop-down box, mapping to a node in the same position as the cells on the "State" overview. Currently these will all be set to "(Ignored)". The background of *pressed* keys will be highlighted in the same way as the cells in the overview; you can see an example in the first row of the third column in Figure 5 on the next page.

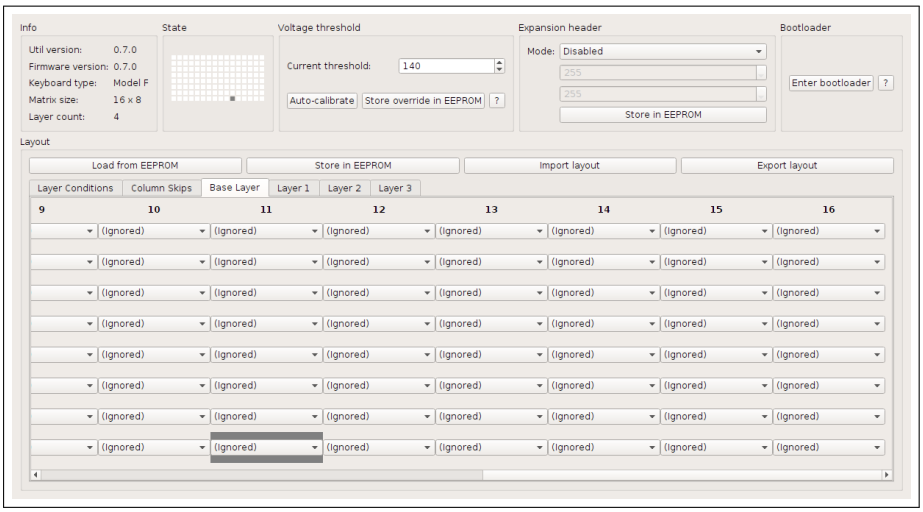

Figure 5: Blank layout

Now, press a key on your keyboard, checking to see where in the overview and on the layout drop-downs the corresponding node responds—you may need to scroll left and right to find it. Choose an appropriate scancode from the drop-down. The new key will take effect immediately; test it out by typing elsewhere.

Repeat this for all of the keys on your keyboard.

6.2.2 Leftover nodes, autocalibration

You will be left with the *pressed* nodes that don't correspond to actual keys, as well as a few *released* nodes that don't have an associated key either.

This is a good thing—if you are careful to set these to either the special "(PRESSED)" or "(RELEASED)" scancodes correctly, the auto-calibration function will be able to operate correctly in most cases. Auto-calibration requires at least one of each type of special scancode to work.

Figure 6 on the following page shows a partial example of a correctly configured PC-AT keyboard.

Once you have these special scancodes set (along with the rest of the keyboard's normal scancodes), try the "Auto-calibrate" pushbutton. If it gives a useful voltage threshold value (if it's different to the value you derived yourself in Section 6.1.3 on the previous page, check to make sure your keys still all register and don't "ghost"), you can choose to have the keyboard auto-calibrate on startup every time.

If the value isn't suitable—some keyboards are stubborn—try playing with the special scancodes (try setting some to "(IGNORED)"). In the worst case, click the "Store override in EEPROM" button to save a forced value, avoiding auto-calibration.

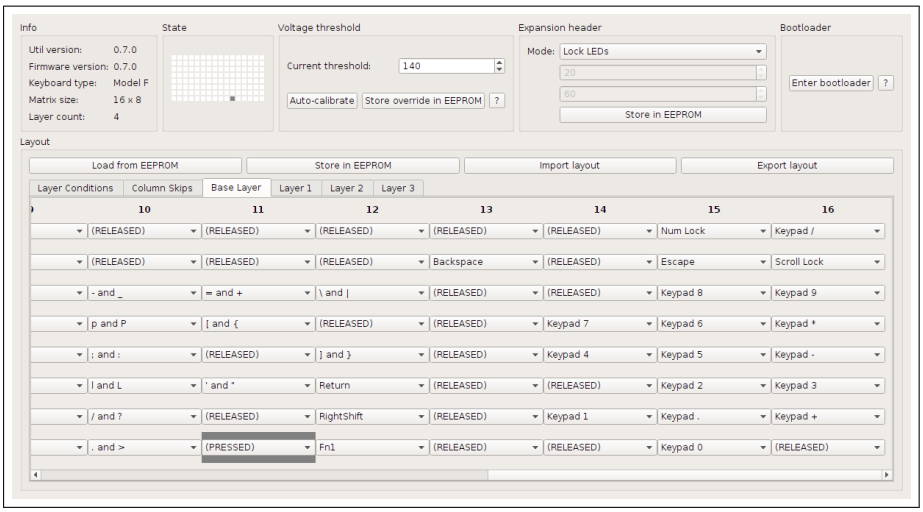

Figure 6: PC-AT with layout set—note "(PRESSED)" and "(RELEASED)" scancodes

6.2.3 Loading/Storing/Importing/Exporting

At the very least, you will want to click "Store in EEPROM" to save the layout you have created to the controller persistently, otherwise the next time you unplug it or reboot it will be lost.

You can also restore the saved layout using "Load from EEPROM" if you have made changes you don't like.

Exporting and importing from disk allows backing up layout configs; the resulting file also contains all of the other user-modifyable settings on the controller, with the exception of the voltage threshold. The file format is in plain text, and can be manipulated with a text editor.

7 Function keys and layers

The firmware supports has three additional layers for extra scancodes. These can be mapped in the same way as the base layer, by clicking on the "Layer (1, 2, 3)" tabs. Be sure to save your settings with "Store to EEPROM" before unplugging/rebooting.

7.1 Layer selection

There are three ways to access subsequent layers:

7.1.1 Fn(1, 2, 3) scancodes

Assigning a key on the base layer to one of Fn1, Fn2 or Fn3 allows accessing layers while they're held. It isn't quite as simple as that, however:

- You must set up a "condition" on the "Layer Conditions" tab; this allows setting up combinations of more than one Fn key—one example is shown in Figure 4 on page 8

- Fn keys are also recognised on subsequent layers; this means you will probably want to choose the same Fn key in the same place on the layer that will be selected when that Fn key is pressed; otherwise the keyboard will rapidly flick between two layers

7.1.2 Select (Base, 1, 2, 3) scancodes

These four scancodes allow selecting a layer semi-permanently without having to hold keys. This is useful for remapping an entire keyboard to an alternative layout, such as Dvorak/Colemak etc.

For example, if "Select 2" is pressed, Layer 2 will be used as the new "default" layer until either the keyboard is unplugged/rebooted, or another Select key (such as "Select Base") is pressed.

Fn keys are still respected. Once they are released, the keyboard will return to the default layer set by the Select key.

7.1.3 Expansion header

The 6-pin expansion header (covered later) can be used with a physical external switch to select certain layers. It works by assigning the external switch to a specified Fn key. The caveats listed in Section 7.1.1 on the previous page still apply.

8 Column skips

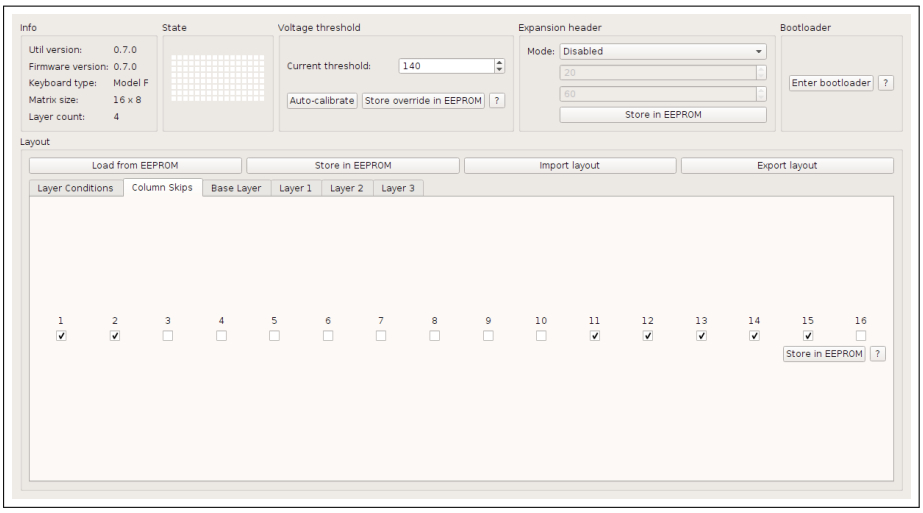

Figure 7: 4704 "Kishsaver"—Columns 1, 2, 11, 12, 13, 14 and 15 skipped

Some keyboards don't use all 16 columns.

There is a scan-rate performance benefit gained by setting the columns to be skipped, as shown in Figure 7. As usual, make sure to use the "Store in EEPROM" pushbutton to make the change persistent.

Skipping columns is an absolute must on certain keyboards that have unused columns tied to ground within the pad card itself. Driving these columns will cause unreliable sensing, and may eventually damage the controller. So far, the known keyboards that do this are 4704 "Kishsavers" and 3178 derivatives, but it is worthwhile remembering this functionality if you are having issues on a smaller keyboard.

9 Expansion Header

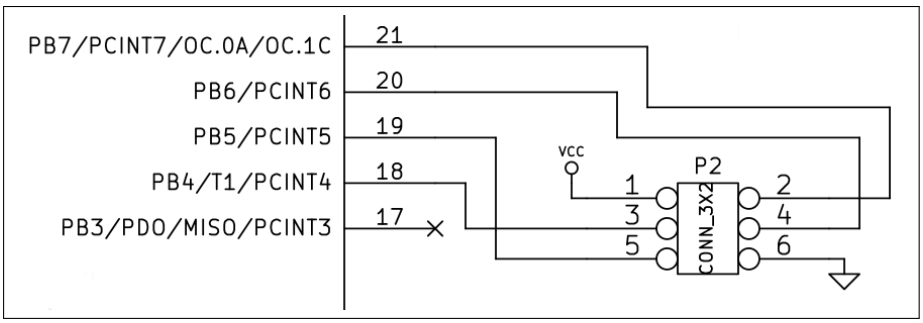

Figure 8: Expansion header pin assignment

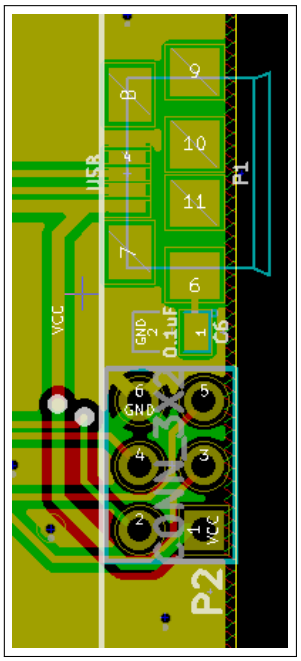

Figure 9: Expansion header on PCB

The expansion header is the 6-pin right-angle header mounted next to the micro-USB socket on the controller (be warned—it isn't an AVR ISP header!).

This presents 5V (VCC), ground, and four GPIO pins from the microcontroller, as shown in Figure 8 on the previous page. Pin 1—VCC—is identified on the controller by being soldered to a square pad, as shown in Figure 9 on the preceding page; this is the lower-right pin when viewing the controller from a normal typing position, or the lower-left pin if viewing the 90° bent pins head-on.

There are are currently three ways to use the expansion header, all selectable via the "Mode" drop-down box underneath "Expansion header" (like before, be sure to store your changes):

9.1 Solenoid/Buzzer

This is primarily designed to interface with the Solenoid-Driver PCB (covered in a separate manual), however can be used with other buzzers, as long as they don't source more than 20–30mA from the output pin (an example is the 4704 Kishsaver's buzzer).

The solenoid will be fired whenever a key is pressed; excluding Fn keys and Ctrl/Alt/GUI keys (shift keys *will* actuate the solenoid).

As keys can be pressed faster than the solenoid can reset in some cases, the controller will queue up to 255 solenoid clicks.

The individual pins on the expansion header are used as follows:

- Pin 2 (PB7) is set high when the controller has finished enumerating over USB, and left on until the controller is unplugged or the PC puts the controller to sleep
- Pin 4 (PB6) is set high when a key is pressed, starting a timer
- When the timer hits the time set in "Extend time (ms)", Pin 4 is set low
- When the timer hits the time set in "Retract time (ms)", Pin 4 will be set high once again if there is another solenoid click queued up, and the cycle will repeat

An example of hooking up a 4704 Kishsaver's buzzer is shown part-way down in this forum post.

9.2 Lock LEDs

No Model F keyboards except for the PC-AT ever came with lock LEDs; however, external LEDs could optionally be wired up using the expansion port.

The pins are assigned as follows:

- Pin 3 (PB4) is assigned to the Scroll Lock LED
- Pin 4 (PB6) is assigned to the Caps Lock LED
- Pin 5 (PB5) is assigned to the Num Lock LED

PC-ATs can be rewired without having to make a new cable by shuffling some pins inside the 8-pin female connector the LED PCB is fitted with. The original IBM mapping—using pin labels from the LED PCB end—is as follows:

A1 Caps Lock LED

A2 Scroll Lock LED

A3 (not connected)

B1 Num Lock LED

B2 GND

B3 VCC

The actual female connector pinout doesn't match this, so be sure to trace through the wires.

9.3 Solenoid/Buzzer + Lock Switch

This is a combination of the Solenoid mode from above, with the addition of a single input from an external switch.

There are a variety of modes here that all work similarly, but with different functions:

- Caps Lock
- Shift Lock
- Num Lock
- Fn(1, 2, 3) Lock

Each one is suffixed with either "NO" or "NC". This refers to "Normally Open" or "Normally Closed"; the appropriate one should be chosen to reflect the configuration of the switch.

The pin assignment is the same as in Section 9.1 on the previous page; however the input is taken from pin 5 (PB5). A pull-up resistor is switched on for pin 5, so the switch should be connected so it will connect the pin to ground when activated.

There are some things to be aware of when using the lock modes:

- When changing between options, the keyboard may end up still set to one of the previous states—you may need to cycle the relevant function (Caps Lock etc.) to get back to a sensible state
- With Caps Lock and Num Lock, the controller only sets the relevant lock when a change-of-state of the switch occurs. This means you can still toggle Caps/Num Lock after you've set the switch
- Fn(1, 2, 3) lock works simply by "holding down" the corresponding Fn key. Take this into account with layer condition combinations, and Select (Base, 1, 2, 3) keys.
- Caps/Num Lock can only work by knowing the current Caps/Lock state on the OS-side; the controller determines this by monitoring the lock LEDs sent over USB (which are still sent by the OS even if you don't have any). If you have these disabled, or set to do something unusual, you may find the Caps/Num Lock switch is unreliable

10 Updating firmware

The controller firmware can be updated over USB without having to open the keyboard. As mentioned above in Section 9 on page 12, the 6-pin expansion header is *not* an AVR programming header—programming is performed exclusively with the USB DFU protocol.

Updated firmware can be either compiled from the source distribution, or downloaded as pre-compiled hex files from http://downloads.cornall.co/ibm-capsense-usb. Firmware is specific to an individual controller type and board revision; make sure to download the right type! Model-F-USB-Rev2 controllers will be named in the format `ibm_capsense_usb_model-f_rev1-2_x.y.z.hex`.

The DFU bootloader can be entered by clicking the "Enter bootloader" pushbutton in the software GUI. Alternatively, in the worst case, the test pads on the underside of the controller can be used to force it into DFU mode; if the PROG pad is shorted out when either RESET pad is momentarily shorted, or the controller is first plugged into USB, it will enter the bootloader.

When updating the firmware, in many cases you will want to update the OS-side utility software to match after the upgrade.

There are two common programs used to program AVR microcontrollers over DFU:

10.1 dfu-programmer

`dfu-programmer` is probably available in your package manager if you're on Linux or BSD.

Pre-compiled executables are available for Windows users (along with drivers) from the project's homepage.

Mac OS X users can install it from MacPorts:

```
sudo port install dfu-programmer
```

Usage is simple (omit `sudo` prefix if on Windows or you have user permissions for the DFU device):

```
sudo dfu-programmer atmega32u2 erase
sudo dfu-programmer atmega32u2 flash hexfilename
sudo dfu-programmer atmega32u2 reset
```

10.2 Atmel FLIP

Atmel's own DFU programming application is available from their website. It is a graphical Java program; versions for Windows and Linux are available.

Be sure to specify the correct microcontroller; the AVR used in the ibm-capsense-usb controllers is an ATmega32U2.

11 Troubleshooting

Contactless capacitive keyboard sensing is more complex than normal switched-contact sensing, and several things can go wrong. Most problems can be solved relatively easily however.

Some other issues surface in relation to layers and other more advanced functionality.

The following provides some pointers; however, assistance is available on the relevant forum threads on Deskthority and Geekhack.

11.1 Unreliable sensing

This is by far the most common problem, and there are several reasons, all which are easily solved. Use the "Status" overview to determine actual sense state to eliminate scancodes and your OS as a problem.

Common causes:

- Your keyboard may have one or more unused column lines tied to ground. This will definitely be the case if it's a 4704 "Kishsaver". Use a multimeter to check—reference Figure 1 on page 3—then use the "Column skips" functionality (Section 8 on page 11) to avoid scanning those columns.
- The ribbon cable hasn't been soldered correctly, or is shorting to something on the controller. Check all connections and look on both sides of the controller for bridged solder.
- The chassis ground connection from the bracket to the controller isn't making good connection; leaving this unconnected leaves the keyboard very prone to electrical interference. Make sure the screw is tight and that the pad is making good contact to the bracket.
- The voltage threshold may be incorrectly set. Before trying auto-calibration, make sure you can manually calibrate it first. Some keyboards won't auto-calibrate well; a manual calibration value can be saved in the EEPROM. See Section 6.1 on page 7 for instructions.
- You don't have a compatible keyboard; this is unlikely unless you have a 3178 or related keyboard, and haven't installed the adaptor (see Appendix A on the following page). Check Section 1 on page 2.
- Lastly—hopefully an unlikely cause—you have mounted the controller upside-down or back-to-front. Check Figure 2 on page 5 to make sure you have installed it correctly.

11.2 Layers

When faced with layer selection issues, first check your firmware version—the firmware should be at least 0.7.0, as layer evaluation has changed significantly starting from this version.

11.2.1 Layers aren't selected when using Fn keys

It is necessary to set up *Layer Condition* (see Section 7.1.1 on page 10) first. Make sure that the mapping between your chosen Fn key points to the correct layer.

11.2.2 Rapid changing between layers with Fn keys

Because the controller evaluates function keys on subsequent layers as it changes into them, it is usually recommended to map a Fn key to the same physical position in both the base layer and the subsequent layer.

If the physical key isn't mapped to the same Fn key in the next layer (for example, it is set to "(IGNORED)" or a normal keyboard key), then the keyboard will see the Fn key, change into the next layer, then on the next scan see that the Fn key is no longer pressed and drop back to the base layer.

11.2.3 Layers aren't selected with Select keys/don't reach expected layer with Fn Lock switch

This requires some care; layer locks (through external switches on the expansion port—see Section 9.3 on page 14) simply simulate holding down a Fn key. Check that the combination of a layer *selection* and the simulated pressed Fn key give the correct resulting layer (remember, Fn keys are evaluated on subsequent layers).

Appendix A: Installation in 3178s and derivatives

The 3178s, despite having the same 30-position 3.96mm pitch ribbon cable, have a very different pinout from the standard Model F keyboards; compare Figure 10 with Figure 1 on page 3. In addition, the matrix is 12x8 instead of 16x8.

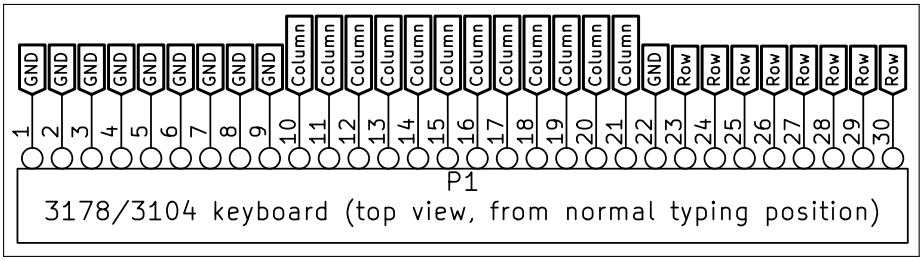

Figure 10: Pinout of 3178/3104 ribbon cable

Because the rows are on the right-hand-side of the ribbon (when viewing the keyboard from a normal typing position) instead of the left, this means the controller must be mounted flipped. In addition, an adaptor is required to match the different pinout; the internal routing of this is shown in Figure 11 on the next page.

An actual Model-F-USB-Rev2 controller with the 3178 adaptor fitted is shown in Figure 12 on page 19. Note how the controller has been rotated upside-down, and the USB cable now exits on the left-hand-side; a chassis ground wire now exits on the right-hand-side.

A.1 Mechanical installation

First, remove the old controller—Section 4.2 on page 4 is still relevant.

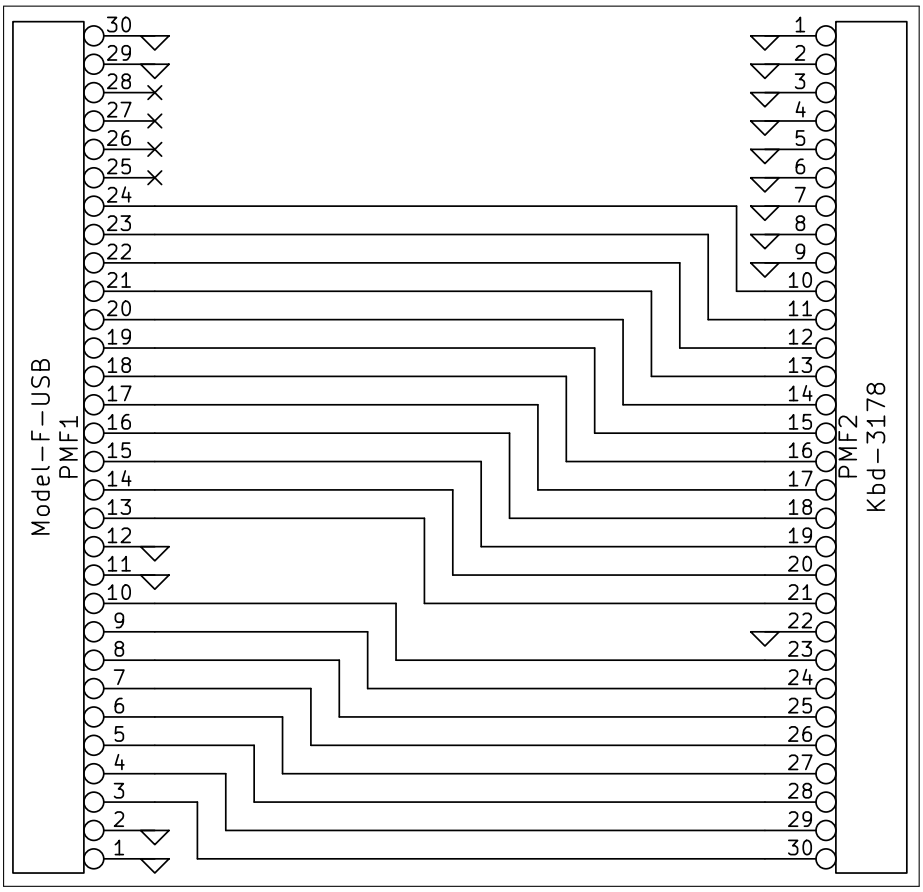

Figure 11: Schematic of 3178 adaptor

A.1.1 Solder ribbon cable

Solder the new controller to the ribbon cable, as described earlier. Make sure the controller is orientated the correct way. The green/yellow ground wire should be on the right-hand side of the keyboard, and the controller should be upside down (you won't be able to read the text/logo). The ribbon cable will enter the adaptor without passing over the controller, and you will be soldering within the white box surrounding the holes on the back of the adaptor (see Figure 13 on the following page).

Make sure the solder joints are solid and there is no bridging between pins.

A.1.2 Attach ground wire

Attach the ground wire to the nearest bracket used to retain the old controller. A good connection here is very important, otherwise the keyboard will be very sensitive to stray electrical noise.

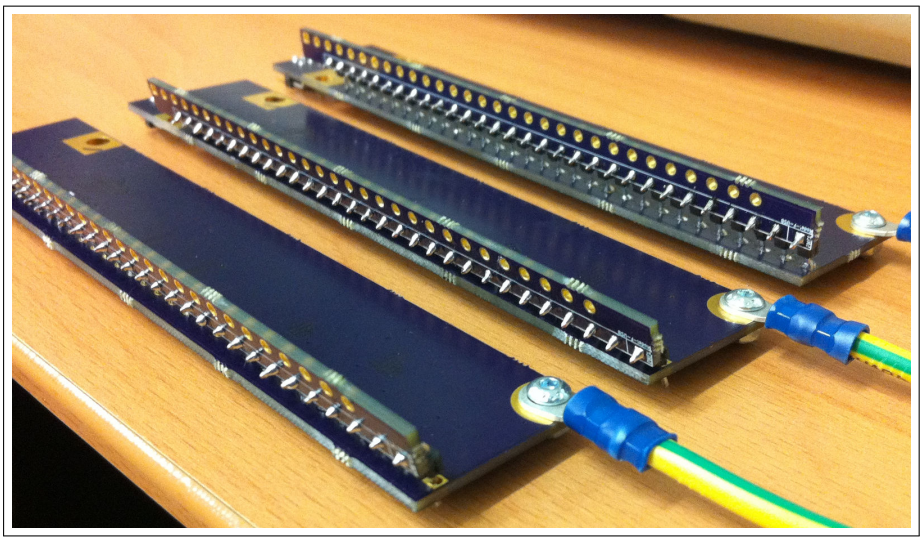

Figure 12: Model-F-USB-Rev2 controllers fitted with 3178 adaptors, showing correct orientation

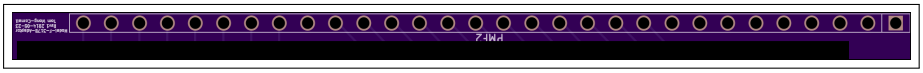

Figure 13: 3178 adaptor, back side (looking over controller); solder within white box, ribbon enters opposite side shown

A.1.3 Mounting

As the mounting screws are in very different positions on the 3178s compared to a standard Model F, it is unlikely that you will be able to attach the controller to one of the brackets. Instead you will have to rely on the ribbon cable and the ground wire to support it. Take great care to check for possible shorts—make sure the controller will not touch anything metallic; you may need to use electrical tape to protect it from its surroundings.

A.1.4 Solenoid and switch

The solenoid can be used with the solenoid driver board; check the separate solenoid driver installation manual for details.

The blue switch from the old controller can be desoldered and used for Caps/Shift/Num/Fn Lock; the supplied ribbon cable with the solenoid driver has two free wires to connect it (blue is 0V, purple is input pin). Mechanical mounting of the switch is not covered in this manual.

A.2 Controller setup

Setting up the controller on a 3178 is largely the same as in Section 6 on page 7. However, you will want to set the controller to skip columns 13, 14, 15 and 16 (see Section 8 on page 11).

You will find that the matrix presented within the software is flipped left-to-right from actual keyboard; this is because the controllre has been flipped left-to-right to cope with the ribbon cable's inverse pinout. This won't affect anything, other than making it slightly unintuitive when mapping the scancodes.