ibm-capsense-usb

Installation: Beamspring-Displaywriter-USB-Rev1

Tom Wong-Cornall
tom@wongcornall.com

July 8, 2014

Contents

1 Compatibility

The Beamspring-Displaywriter-USB-Rev4 is, as far as has been determined so far, compatible with all IBM Beamspring Displaywriter keyboards, and possibly the 5253 and 5254 Display Station keyboards (*not* the 5251 and 5252—these are "standard" beamsprings), which appear to be basically Displaywriter keyboards with dark keys.

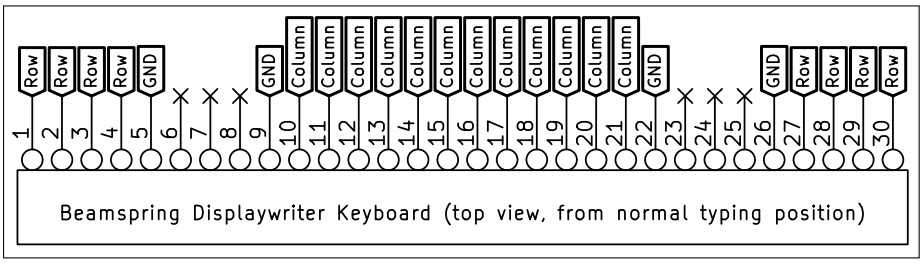

Figure 1: Pinout of Displaywriter edge connector

In general, as long as the keyboard has a 3.96mm-pitch 30-position edge connector, and conforms to the pinout in Figure 1, then it should work. Keyboards that have less than the 12 columns and/or 8 rows are still compatible; the missing keys can just be ignored.

2 Required parts and tools

In addition to the Beamspring-Displaywriter-USB-Rev1 controller, you will need a mini-USB cable.

Nothing more than a medium flat-head screwdriver should be needed in most cases for installation. For dirty keyboards, or keyboards that have a disintegrating contamination shield, some additional tools (such as mini-vacuums) may be required to remove the bottom-plate from the keyboard itself and clean the pad card.

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 beamspring 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 before removing the original controller.

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

Take *great* care if dismantling the beamspring keyswitches themselves. The beam springs themselves can be exceptionally fragile in some cases if the keyboard wasn't stored well, and some people have snapped the spring steel when taking them apart. It hardly needs pointing out that IBM no longer manufacture spares!

Also be careful if you find it necessary to remove keycaps. The stems are simple flat sections of stamped steel, and the corresponding mount inside the keycap is weaker than a modern Cherry MX mount. Never rock the keycaps side-to-side; pull them up and tilt them only forwards and backwards.

3.3 Cabling

Don't kink or fold the mini-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.

3.4 Short circuits

Ensure the controller won't come into contact with any metal parts or wires that could cause shorts. When plugged in with the edge connector, it should be clear of the top-plate (which is normally covered with a clear rubbery material anyway). Make sure the chassis ground wire is securely connected so it won't come loose and touch other exposed wires or pins.

3.5 Cleanliness

Capacitive switching is very sensitive and deals with tiny signal levels. Dirt or other contamination (ironically, the rubber contamination shield can be a source of trouble) 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 top half of the case can be removed with four or so screws on the underside of the keyboard. In many cases, no further disassembly will be required.

If, however, you are confronted with the sort of disintegrating contamination shield pictured in Figure 2, further work will be required; see Section 4.3 on the next page.

4.2 Removing old controller

First remove the cables. The main communication cable will attach to the original controller via a clip-in connector with some pin headers.

Figure 2: Original Displaywriter controller, image courtesy of webwit

In addition, there may be a speaker, attached via two twisted wires to a pin header. These can simply be pulled off.

The old controller will be held in with two or more screws attaching it with brackets to the top plate. Remove these, then gently slide the entire controller backwards off the protruding edge-card contacts of the pad card.

The old cable can be removed now. The bottom-half of the case may need to be removed from the keyboard itself (shown in Figure 2 on the preceding page. It will normally be fastened via four screws on the far left and right sides of the keyboard. When removing the old cable, be careful to retain the strain relief, as it may come in useful for retaining the mini-USB cable.

4.3 Decontamination

The beamspring keyswitches in themselves have no special protection from spilled liquids, and are not self-draining. To counter this, IBM fitted a flexible rubber shield between the keyboard and the keycaps to protect the board.

Unfortunately, rubber degrades with age and many contemporary keyboards will suffer from a contamination shield that is flaking apart (depicted in Figure 2 on the previous page). This will eventually find its way inside the top/bottom plate sandwich and can cause sensing issues; keys may flicker on/off by themselves, or won't be reliably picked up.

If this is the case, it is best to remove the contamination shield entirely. This involves removing all of the keycaps and carefully lifting the thin sheet off. In some cases a simple vacuuming of the leftover remnants may be enough. If, however, fragments have made their way inside the keyboard, the only recourse is to clean this too:

1. Remove the keyboard from the case bottom if you haven't already
2. Support the "keyboard sandwich" at the sides with some books or similar to ensure no keys are pressed
3. Remove the screws holding the bottom-plate, then remove the bottom-plate, taking note of where the screws came from (not all holes are threaded)
4. Remove the "pad card" (internal PCB), being careful not to disturb the keyswitch modules. A clear rubber insulator sheet may peel off as well—put this to one side
5. Clean all fragments of contamination shield, dust and dirt from the pad card
6. Reassemble, tightening the screws gradually in a figure-of-eight pattern

Some instructions from one of the original IBM manuals are available at http://webwit.nl/input/ibm_beam_spring/manual1.gif.

5 New controller installation

5.1 Mechanical installation

1. Route the mini-USB cable. If possible, make use of the original strain relief. Leave enough slack so the cable will almost meet the protruding edge connector of the pad card
2. Plug the new controller into the pad card; orientate it so the text is visible and readable from a typing position
3. Make sure the contacts on the pad card are correctly aligned with the contacts inside the connector (it is possible to slightly misalign these from left-to-right)
4. Some IBM keyboards have a raised edge to the right-hand-side of the pad card, preventing the new controller being seated adequately (most don't cause it to lose contact, however). You may need to trim some material from the blue plastic connector if it seems you will not make good contact. Be careful not to damage the internal metal contacts.
5. Connect the "Chassis GND" wire to the nearest screwhole using one of the screws used to retain the old controller; make sure the screw is snug and the ring terminal makes good contact
6. With the PC-end disconnected, plug in the USB-mini cable
7. Plug in the PC-end of the USB-mini 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.

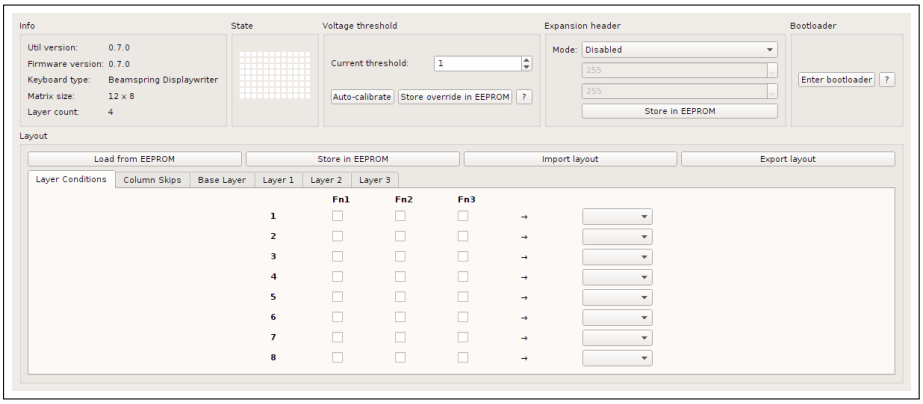

Figure 3: First startup of GUI util

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 1. 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 96 cells—12 wide and 8 high—represent the sensed state of each node in the keyboard matrix. With a voltage threshold of 1, 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 raise the threshold, stopping to press keys now and again to see if they register. The maximum threshold is 1023, but you will probably find the keys will correctly register somewhere within a wide band between 100 and 200.

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 these calibration nodes registering.

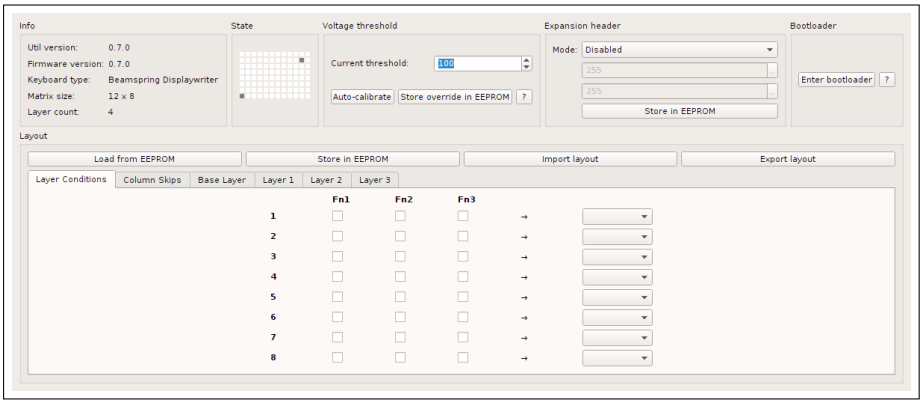

Figure 4: 3727 keyboard with sense threshold correctly set

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.

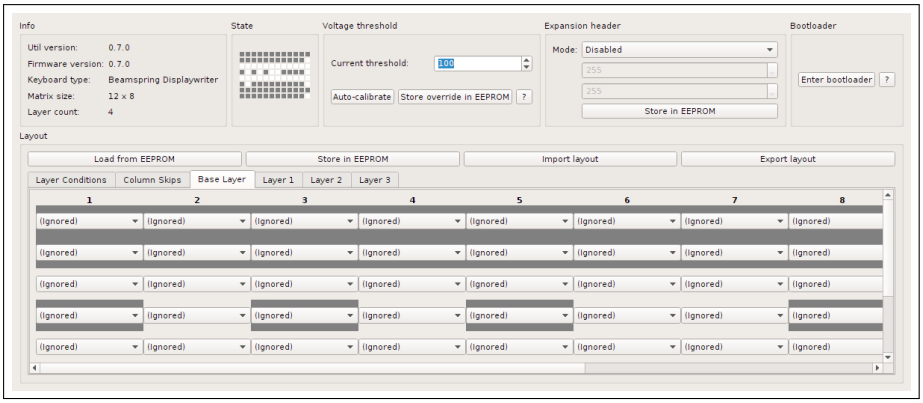

Figure 5: Blank layout

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 next page shows a partial example of a correctly configured (standard beamspring) 3727 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 preceding 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.

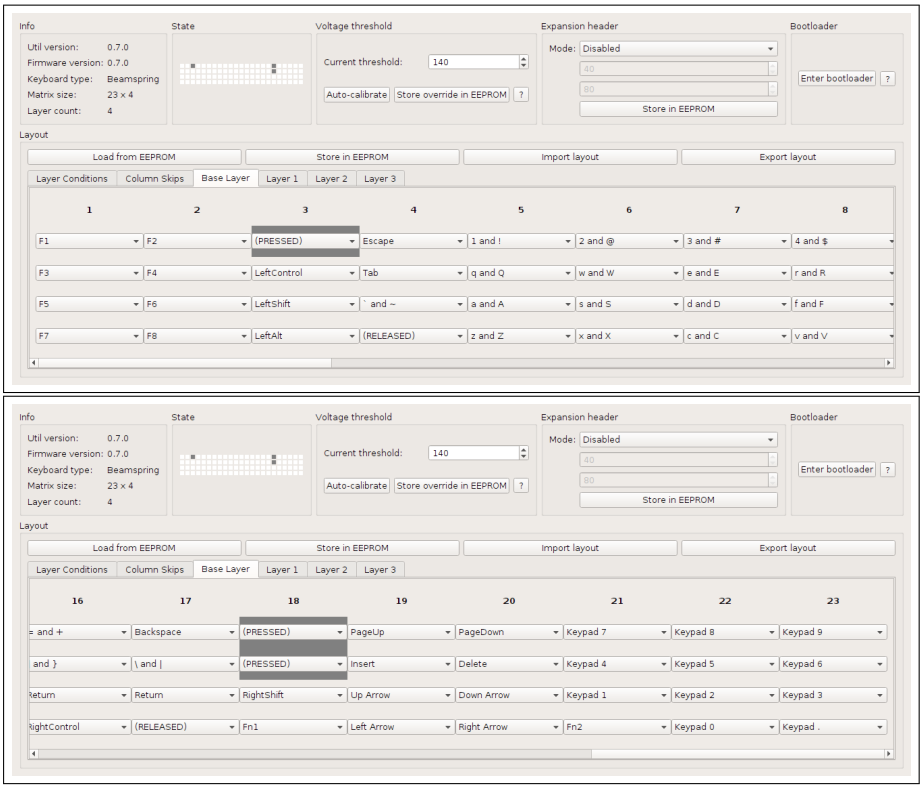

Figure 6: 3727 with layout set—note "(PRESSED)" and "(RELEASED)" scancodes

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 still apply.

8 Column skips

Some keyboards smaller keyboard may not use all 12 columns. In many cases, the scancodes in the unused columns can simply be set to "(IGNORED)" or "(RELEASED)".

There is a scan-rate performance benefit gained, however, by setting the columns to be skipped, as shown in Figure 7 on the next page. 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 only known keyboards that do this are Model F keyboards such as the Kishsaver, but it is worthwhile remembering this functionality if you are having issues on a smaller keyboard.

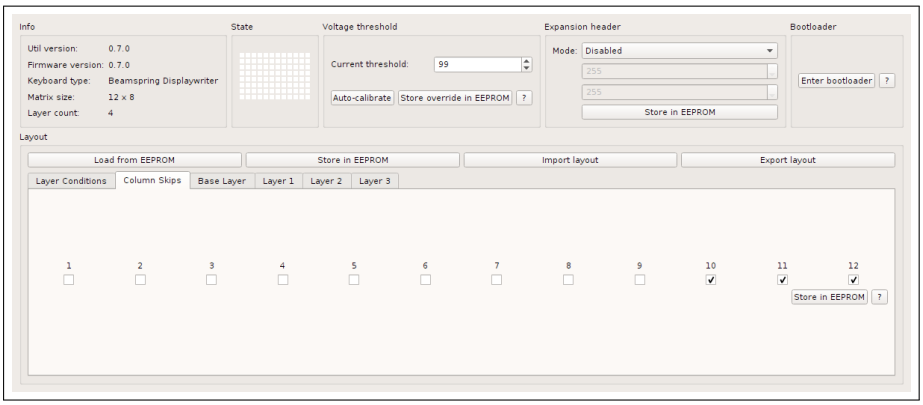

Figure 7: Columns 10, 11 and 12 skipped

9 Expansion Header

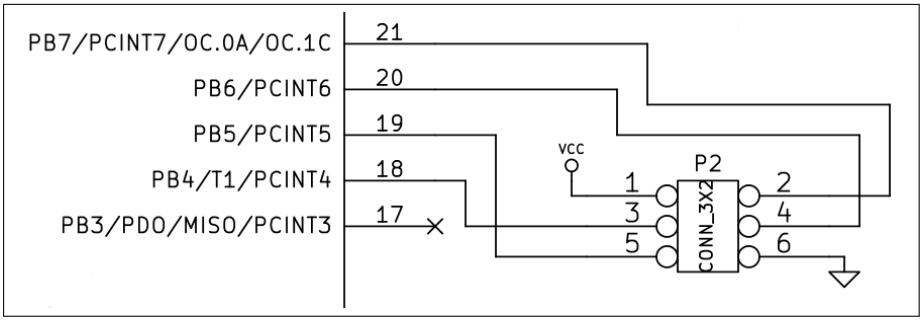

Figure 8: Expansion header pin assignment

The expansion header is the 6-pin header mounted next to the main microcontroller chip 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. Pin 1—VCC—is identified on the controller by being soldered to a square pad, as shown in Figure 9 on the next page; this is the lower-left pin when viewing the controller from a normal typing position.

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 Model F Kishsaver's buzzer).

Be warned that the 8-ohm speaker normally fitted to Displaywriters cannot be directly wired to the expansion port, as it will demand too much current

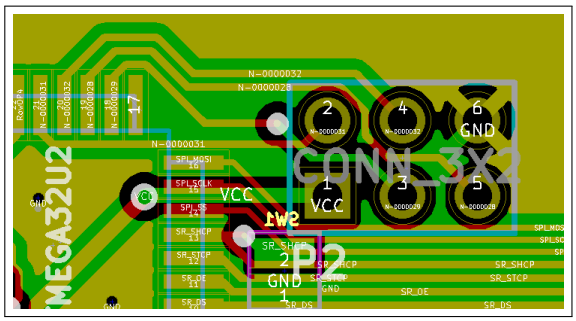

Figure 9: Expansion header on PCB

and would damage the microcontroller. Instead, an external circuit such as the Solenoid-Driver PCB must be used to interface between.

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

9.2 Lock LEDs

Although no Displaywriter keyboards ever came with lock LEDs, 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

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 page 12; 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! Displaywriter Rev1 controllers will be named in the format `ibm_capsense_usb_bs-beamspring_rev1_x.y.z.hex`.

The DFU bootloader can be entered by clicking the "Enter bootloader" push-button 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:

- The edge connector isn't making good contact with the pad card—make sure the controller is seated correctly (some keyboards have such a large protrusion on the right-hand-side that a slot must be filed into the edge connector).
- The edge connector is misaligned with the contacts on the pad card—look closely to make sure the connector contacts are centred on each pad card contact.

- The chassis ground wire isn't hooked up or isn't make good contact; leaving this unconnected leaves the keyboard very prone to electrical interference.
- The contamination shield may have deteriorated and left fragments inside the keyboard itself; see Section 4.3 on page 5 for instructions on cleaning.
- 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.
- Your keyboard may have one or more unused column lines tied to ground. Use a multimeter to check—reference Figure 1 on page 2—then use the "Column skips" functionality (Section 8 on page 11) to avoid scanning those columns.
- You don't have a compatible keyboard; this is unlikely unless you have a standard beamspring (non-Displaywriter) keyboard. Check Section 1 on page 2.
- Lastly—hopefully an unlikely cause—you have mounted the controller upside-down. Make sure the controller is in a position where you can read the text/logo on it when you are in a normal typing position.

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 11) 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 13) 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).