BCR-2000 Control Surface Software for Software Audio Console (SAC)

Disclaimer:
This software is provided "AS-IS" with absolutely no warranty of any kind. This software is "alpha-test" software and may contain critical defects (software bugs). Neither Randall Hyde, nor Plantation Productions, Inc., assume any liability for the use of this software. You should strongly consider not using this software in its current alpha-test state for any mission-critical applications (such as mixing a live show) until you have rigorously tested it and found it suitable for your own application.

Introduction:

SACCS (Software Audio Console Control Surface) is a small Windows application that allows you to connect a Behringer BCR-2000 control surface to a SAC system. It provides physical controls for many items on a SAC channel strip.

SACCS provides the following features:

1. It provides physical controls (knobs and buttons) for most of the functions found in the SAC Wide Mixer window.
2. It does not require any "overloaded" controls (that is, each knob or button on the BCR-2000 serves a single purpose, you do not have to be any a special mode or have to press any other buttons to put the BCR-2000 into a special mode to use the supported knobs and buttons).
3. SACCS is laid out in a ergonomic fashion. However, if you prefer it's each to change the layout of the knobs using the BC Manager program (available as a separate, free, download).
4. SACCS responds to MIDI control change commands, it's very easy to program other control surfaces to adjust SAC parameters.

Some drawbacks:

1. The BCR-2000 provides only 32 knob controllers, which is about a dozen too few to completely support all the controls in SAC's Wide Mixer view window. A future version of SACCS will eliminate this problem by providing support for two BCR-2000 units or a BCR-2000 and some other unit (such as a KORG nanoKontrol).
2. The Aux send knobs could be layed out better if two BCR-2000s were available. Would also be able to support the Aux Send pan controls with two BCR-2000 units.
3. MIDI setup in SAC has always been complicated, SACCS increases the pain a little.
4. You will need to obtain two other pieces of software (both free these programs at the following URLs:

- BC Manager: http://home.kpn.nl/f2hmjvandernberg281/bc2000.html
- MIDI-yoke: http://www.midiox.com An alternative to MIDI-Yoke is the "Maple Virtual MIDI Cable" (http://www.maplemidi.com). Supposedly this is more stable than MIDI-Yoke, though I haven't had any problems with MIDI-Yoke *except* right after I first installed it (could have been doing something wrong).

While you're at the MIDI-Yoke site, you should pick up a copy of MIDI-OX, too (note that there is a $50 license fee for MIDI-OX if you use it in a commercial environment; playing around with it at home doesn't require $50, but if you use it at gigs (not required by SACCS), you should pay the $50.)

5. You will need to reprogram the BCR-2000 with the sysex file I provide (thus borking it for other uses). Note that the BC Manager download is the software you can use to upload my sysex file to the BCR-2000. You can also use any other sysex download program that works with the BCR-2000, though BC manager, once you get past its skimpy documentation, it quite good.
6. I don't program the LEDs on the BCR ring encoders because I find them distracting. I wish I could program the LEDs on the buttons, but SAC, apparantly, doesn't support this with the Mackie MCU MIDI controller template on which I've based SACCS.
7. I've yet to figure out a way to make the LEDs on the BCR-2000 buttons stay off. They toggle every time you push them. This means that everytime you change the hot channel in SAC, the LEDs will probably be grossly out of phase with reality. Ignore the LEDs. Look at the Wide Mixer view (which is what you should be doing anyway).

Setting up SACCS

First, download the zip file (from http://webster.cs.ucr.edu -- follow the SAC link) and unzip it. This archive contains the following files:

1. SACCS.bc2 (sysex file in BC Manager format)
2. SACCS.syx (sysex file in MIDI format)
3. SACCS.exe (SACCS executable program
4. MidiCtrl_BCR2000.mct: This is actually the MidiCtrl_Mackie_Control.mct file that has simply been renamed; SACCS uses the Mackie MCU template as its base template. I've renamed it to MidiCtrl_BCR2000.mct just to help avoid confusion.

If you haven't done so already, download the BC Manager and MIDI-Yoke packages and install them on your computer. MIDI-Yoke is simply a device driver and will have no user interface.

Before downloading my sysex file, make sure the BCR-2000 is in USB-1 or USB-4 mode. Once you download the sysex file I provide, you won't be able to make this change (see the Behringer documentation for more details).

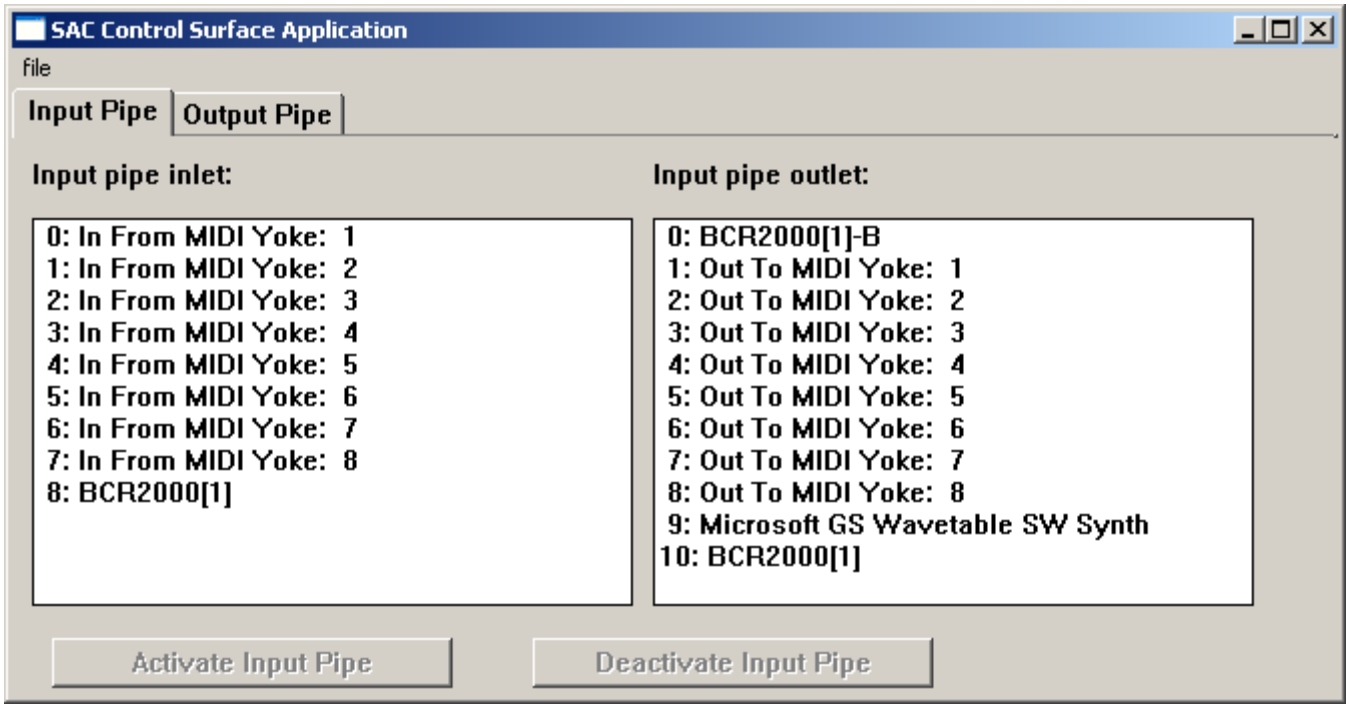
Run BC Manager, initialize it appropriately to recognize your BCR-2000. At this time, it would be a real good idea to copy the data from the BCR-2000 to BC manager and save the data as a backup (the standard Behringer BCR-2000 sysex file is available for download from Behringer, so if you completely mess things up you can always grab the original file from their website and restore your BCR-2000 to its factory condition).

In BC Manager, load my SACCS.bc2 (or SACCS.syx) file and then use the MIDI->Send All Data menu item in the B-Controls window to download the sysex file to the BCR-2000.

Okay, you're done using BC Manager. At a later date, you can user BC Manager to reconfigure the layout of the knobs, if you like, but that's purely optional.

Note that my sysex file on the BCR-2000 disables the store, learn, edit, exit, <- (preset) and -> (preset) keys. From my perspective, the BCR-2000 is dedicated to serving SAC and I don't want to allow it to get into some funny mode (e.g., a different preset) if a user accidentially hits one of these buttons. You can use BC Manager to reactivate these buttons if you don't share the same user interface design philosophy as I do (or if you're forced to use the BCR-2000 for multiple purposes and you don't have any extra money laying around to buy a BCR-2000 to dedicate to SAC).

Okay, run the SACCS.EXE program. This should produce the following window:



SACCS is a *MIDI filter program*. You insert it between a MIDI device (e.g., a Behringer BCR-2000) and another application that would normally receive MIDI input from that device (e.g., SAC). Data that would normally go from the device to the application is first routed through SACCS which does various transformations on the data before passing it along. In particular, the BCR-2000 sysex file I provide programs all the knobs and buttons on the BCR-2000 with very generic control-change and MIDI note values that have no meaning whatsoever to SAC. However, SACCS recognizes these values, transforms them into values SAC does recognize, and then passes those values on to SAC.

So why bother with SACCS? Why not simply program the BCR-2000 to emit codes that SAC directly understands? If SAC provided a generic MIDI template that provided complete access to all the user-manipulated controls in SAC, this would be a possibility. Unfortunately, SAC does not have such a MIDI template. The closest MIDI template SAC provides with complete control is the Mackie MCU control sur-

face; unfortunately, SAC "overloads" most of the SAC functions on eight rotary encoders ("VPots"). The same MIDI command to SAC does different things depending upon what "mode" the Mackie MCU is in. That's were SACCS comes in -- it keeps track of the current mode (or switches to a known mode when the current mode is unknown) before sending the VPot commands to SAC. This allows the 32 different rotary knobs on the BCR-2000 to function like a sequence of button presses and 8 knob turns on the Mackie unit.

SACCS provides two *MIDI pipelines*: an input pipeline and an output pipeline. The terms "input" and "output" can be confusing because both pipelines have an input (or *inlet*) side and an output (or *outlet*) side. To keep things straight in your mind, just note that this document always uses the phrases "input" and "output" in a SAC-relative form. That is, an *input* is an input to SAC and an *output* is an output from SAC.
Currently, the BCR-2000 interface only uses the SACCS input pipe. The output pipe is active (and simply passes information from its inlet to its outlet), but it currently isn't used by SACCS.

To use SACCS, you must select the "Input Pipe" tab in the application's window. On this tab you will see two Windows' list boxes labeled *Input Pipe Inlet* and *Input Pipe Outlet*. You must connect the SACCS input pipe inlet to the BCR-2000 device (it's named *BCR2000[1]* in the example I used above, but this can vary depending on which device driver you install or use for the BCR-2000). You must also connect the outlet of the pipe by selecting an entry from the second list box. Generally, this will be one of the MIDI-Yoke devices, most likely *Out To MIDI Yoke: 1* (unless you are using that MIDI-Yoke device for some other purpose). The exact MIDI-Yoke device you select is unimportant other than you must remember the name and supply it to SAC.

Once you select an inlet and outlet for the input pipe, the *Activate Input Pipe* button will become enabled. You must press this button in order to start processing input data from the BCR-2000 through SACCS. Note that you will need to deactivate the input pipe if you want to change the inlet/outlet assignments. Once you activate the pipeline, the *Deactivate Input Pipe* button is enabled (and the *Activate Input Pipe* button is disabled). You can stop MIDI processing by pressing the deactivate button. You can restart it (possibly after changing inlet/outlet assignments) by again pressing the inlet button.

That's all there is to using SACCS. The Output pipe isn't used to support BCR-2000 to SAC communications. You can play with it all you want, it doesn't do anything (yet, this may change in a future version) other than pass the data from its inlet to its outlet.

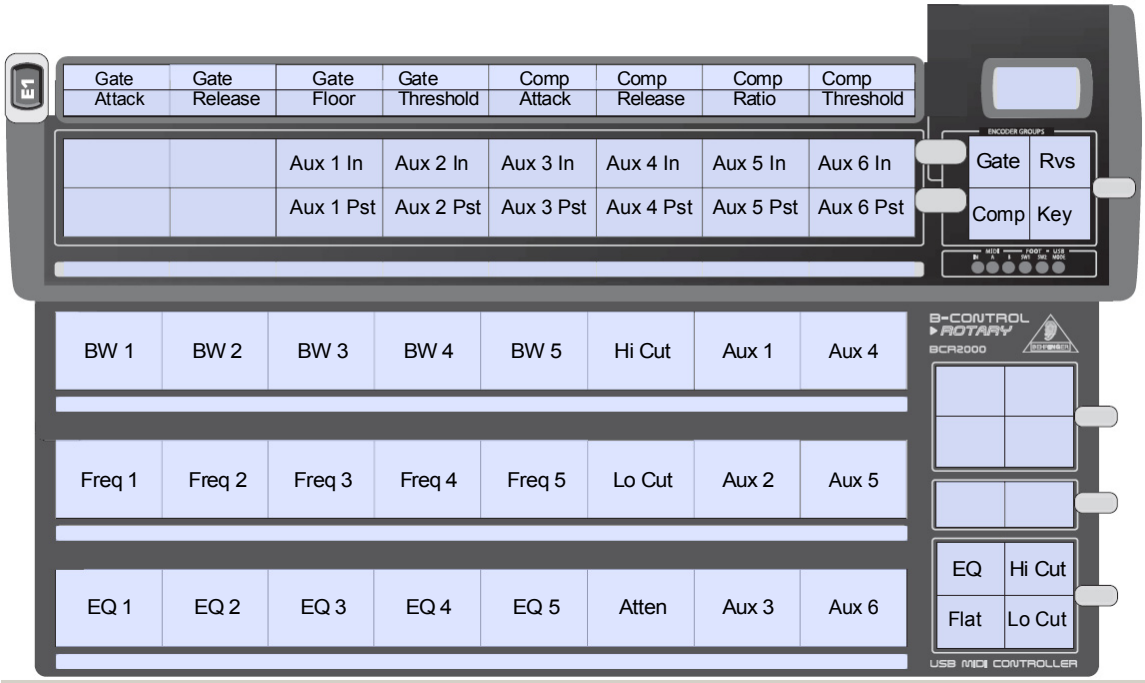
Getting SACCS up and running is only part of the initialization process. You also have to activate the MIDI control surface within SAC itself. Currently, here are the steps you'll need to follow to achieve this.

1. In the options menu in SAC, select *MIDI Device Setup*. This should bring up the following window:

2. If you press on the pull-down menu box next to Midi-Control Device In, you should see a menu pop up that lets you select from the various MIDI devices installed on your system. You will see all the MIDI-Yoke devices ("In From Midi Yoke: 1" through "In From Midi Yoke: 8") as well as the Behringer device and any other MIDI devices on your system. Select the "In From Midi Yoke: 1" device, assuming that you specified "Out to Midi Yoke: 1" as your SACCS outlet device earlier. If you selected some other Midi-Yoke output device, select the corresponding Midi-Yoke input device here.
3. For BCR-200 control, you don't have to select up a Midi-Control Device Out device; just leave that entry as "Not Assigned."
4. Select the File->Midi-Control Template File->Open menu item. This will open up a dialog box. From this dialog box select the MidiCtrl_BCR2000.mct file provided with the SACCS zip file. If you don't see this file, it's probably because you didn't copy it to the SAC\Configuration directory (oh, did I forget to tell you to do that?). In that case, you can also select the MidiCtrl_Mackie_Control.mct file, which is exactly the same file (I just renamed it to provide a BCR-2000-specific filename people could refer to).
5. Select the Smpte/Midi->Midi-Control In Active menu item to activate the BCR/SACCS Midi input.
6. Select Mixer->Navigation Links->Link Midi Hardware Controller to Wide Mixer View so that the BCR-2000 control set always applies to the Wide Mixer (it won't work anywhere else).
7. You should be in business. Hit "F2" or otherwise bring up a Wide Mixer view and you should be able to twist knobs on the BCR-2000 and modifier Wider Mixer view parameters to your heart's content.

The following figure shows the BCR-2000 layout for SACCS. Note that the Encoder Group buttons are simple buttons; in keeping with my philosophy of "no overloaded controls" I do not create four encoder groups as is typical for Behringer BCR-2000 templates. Instead, the encoder group buttons are simply four additional buttons (that select Gate on/off, Gate Reverse, Compressor on/off, and Key Listen on/off).

Several of the buttons on the template are unused. I do have some future plans for some of these buttons, but largely I ran out of SAC functions to put on them.



The top row of encoders implements the dynamic section functions (gates and compressors).

The two rows of buttons immediately below the dynamics section implements the Aux in/out and Aux pst (pre) operations. The first four buttons are unassigned. I will assign them to other functions in SACCS v2.0 (when I support a second controller). For now, they do nothing but toggle their LEDs when you press them. Note that the LEDs on the Aux in/out and Aux pst (/pre) buttons toggle when you press the buttons, but this does not necessarily reflect the state of the actual SAC function. Always look at the Wide Mixer screen to determine the Aux in/out and pst state.

As noted above, the gate, Rvs, Comp, and Key buttons on the BCR-2000 activate/deactivate the gate and compressor, and turn on/off the gate reverse function and key listen functions. As for all the other buttons, the LEDs on these buttons toggle when you press them, but they do not necessarily reflect the state of that function within SAC.

The first five columns of rotary encoders in the bottom section of the BCR-2000 for the EQ section. Each column corresponds to parametric EQ #1 through #5 in the Wide Mixer view. The three rows (top to bottom) provide the Bandwidth (Q), Frequency, and Gain functions.

The sixth column contains the Hi Cut, Lo Cut, and Attenuator functions (yeah, bad spot for the attenuator, this may change in SACCS v2.0; it was the only encoder available after reasonably placing all the other functions)

The last two columns of encoders let you set the Aux 1..Aux 6 send values.

Note that SACCS does not provide a controller for the channel gain or pain. Nor will SACCS ever do this. My assumption is that you will be using a BCF-2000 (with faders) for channel gain control and panning (running in a different instance of SACRemote; more on that later). The BCR-2000's knobs are not the appropriate controllers for channel gain, so I didn't bother here.

There are several missing functions on the BCR-2000. Examples include input channel selection, output channel selection, X-Y Panning, Aux panning, and some other minor stuff. Some of the omissions exist because SAC doesn't support that functionality through the Mackie MCU template (on which SACCS is based). Some functionality isn't provided because I simply didn't have enough controls to go around (without overload, which I refuse to do). Most of the functions, except Channel Gain and Panning (which belong on a different controller) are rarely-used functions and can be handled on the screen with a mouse when they're really needed. In SACCS v2.0 I'll support two BCR-2000 controllers and I can add the other functionality that is supported by SAC and the Mackie MCU template.

Enjoy,
Randy Hyde
President, Plantation Productions, Inc.