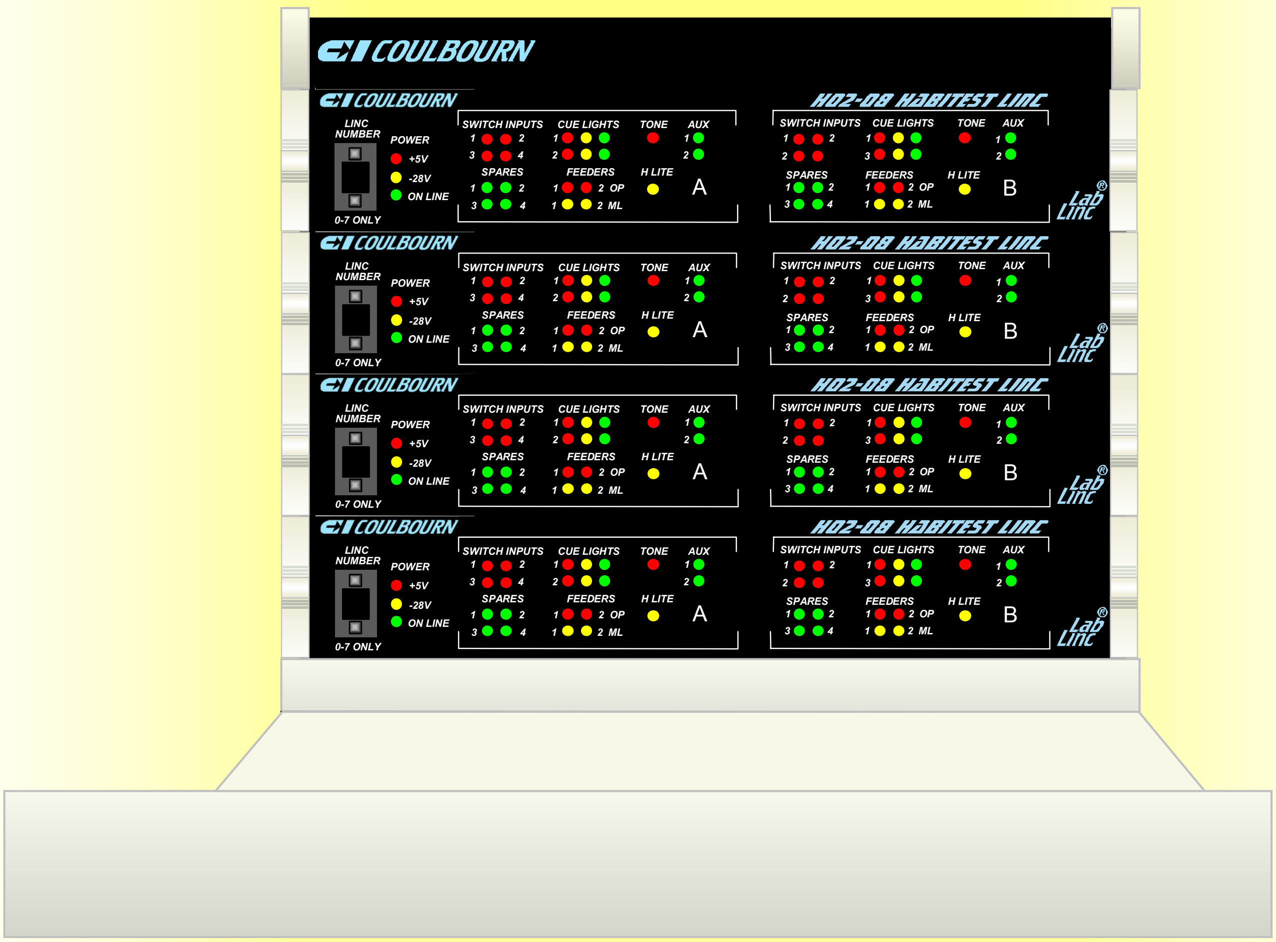
Graphic State 2

A Tutorial

Revised September 12, 2003



CE

COULBOURN INSTRUMENTS

Table of Contents

Where to find information

The software CD that came with your system contains

1. installation instructions
2. driver software for the Coulbourn PCI card
3. the Graphic State Notation software

Both pieces of software will install under the control of InstallShield. Follow the installation order that is described in the instructions on the CD carefully (first install the PCI driver software, then physically install the card and reboot to allow Windows 98/ME (2000/XP compatibility coming soon) to find the hardware and the driver, and finally, install the Graphic State Notation software).

The other CD that came with your system contains all of Coulbourn's technical manuals. These are also available for download from the Coulbourn website (www.coulbourn.com). The two you are interested in now are *The Habitest System H-Series User's Guide* and *Graphic State Notation Software User's Guide.*

The H-series guide describes the setup of the Habitest hardware. This includes the control system hardware (the power base, one or more Lincs with Environment Control Boards and cables) and the environmental hardware (cages, hubs, runways, stimuli, response sensors). The Graphic State Notation guide describes the software.

Both of these are excellent reference manuals, but they may be a bit overwhelming when you first face an array of new equipment. This manual is designed to be a tutorial guide to getting the equipment set up and working with your first Graphic State Notation protocol.

What is a state notation language?

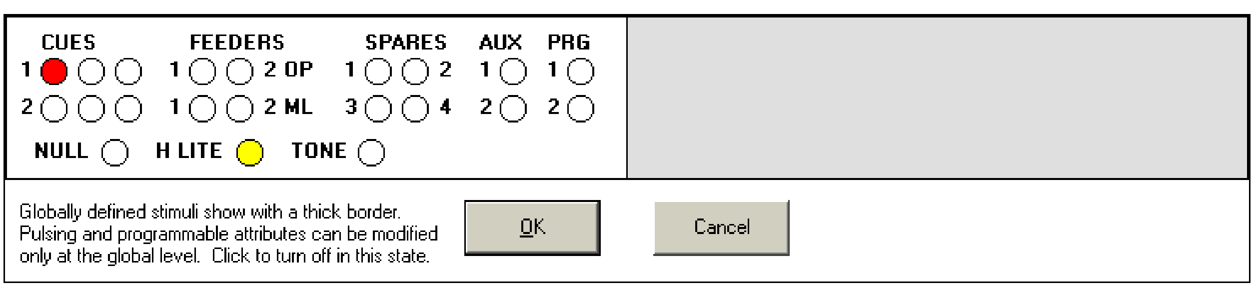
There is a history of state notation languages in the field of animal behavior research. Two of the earlier state languages were SKED and ACT.

In Graphic State Notation, as in these predecessor languages, an experimental protocol is specified as a series of states through which an experiment moves. Each state specifies a stimulus configuration in the subject's environment and a set of time and/or response requirements that cause the program to exit the current state and move on to another state. One of the innovations that Graphic State Notation brings is that these time and/or response requirements may be easily carried across multiple states, included in some and excluded from others.

Graphic State Notation is a "point and click" program that presents options to be selected by the user and represents the structure of each state graphically.

How is a state defined?

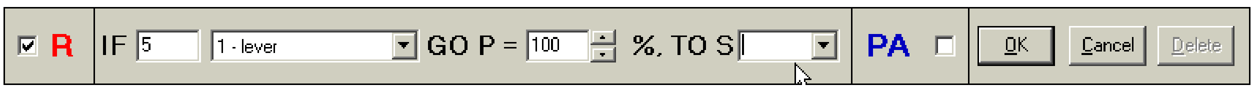
As you will see in the many examples in this manual, a state is defined by giving it a number (and, optionally, a descriptive title) and then clicking on the buttons on a graphical panel of available stimuli to select the ones that should be turned on in the state. In this example, the first red cue light and the house light have been selected.



How is state flow controlled?

There are three ways to exit from a state – IF a specified number of responses have been detected on a particular input switch, AFTER a specified period of time has elapsed, or UPON the nth attempt to enter the state. Each of these will be explained in detail in the examples later in this manual.

These flow control statements are added to a state by filling in the blanks in pop-up forms. This example shows an IF statement partially completed.



What data are collected?

All connected switch inputs are recorded (and time stamped) whether they are involved in controlling state flow or not. All of the onsets and offsets of each switch are recorded. In addition, every state transition is recorded.

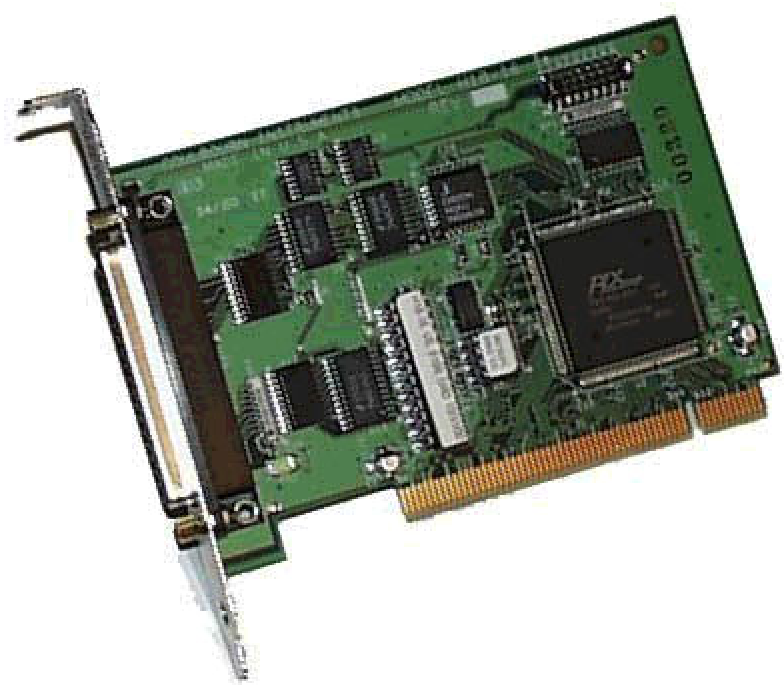
There are many data analyses that can be selected – either during the initial definition of a protocol or at any time later, even long after data have been collected. In addition, the raw data containing a record for every switch onset, switch offset or state transition can be exported at any time for further processing.

How does Graphic State communicate with the experimental environment?

Details about much of this can also be found in the Habitest User's Guide. A brief overview is presented here.

The PCI interface card in the computer

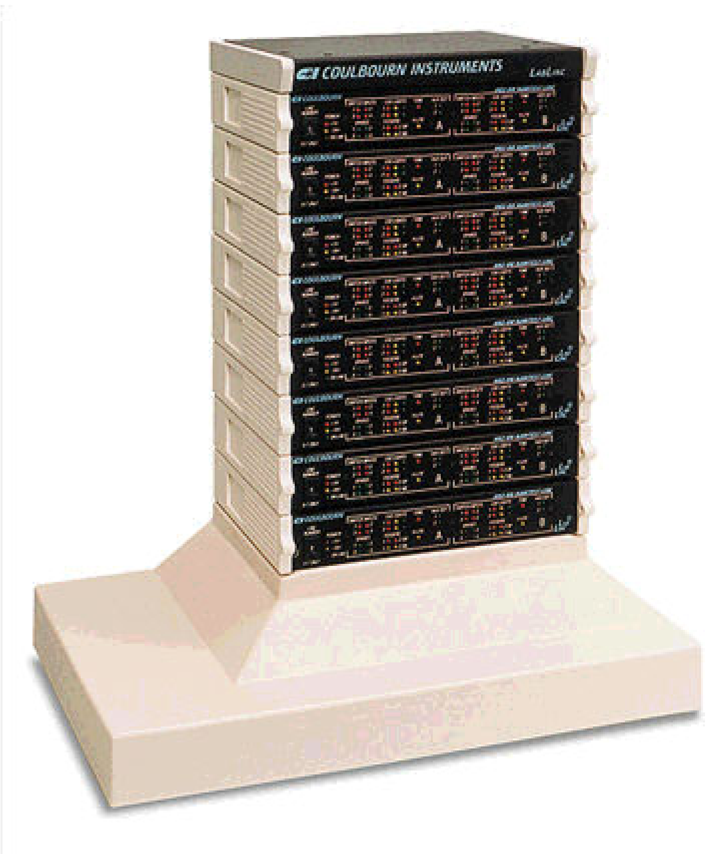
A critical part of the communication interface is the Coulbourn PCI card that should be installed in any available PCI slot in your PC. There are three parts to this installation – installing the Coulbourn PCI driver software, physically installing the card in your computer, and having your Windows 98/ME (2000/XP compatibility coming soon) recognize the new hardware and associate the appropriate driver with it. These three things should happen in the order listed. There are instructions on the CD you received with the software on it that guide you through these three steps.



The Graphic State software talks back and forth through the PCI card to all of the Habitest Lincs that are installed in a stack on your power base. There is a computer cable, supplied with the power base, which connects to the PCI card and to the base.

The power base and the stack of Habitest Lincs

Your Habitest power base supplies power to one or more (up to eight) Habitest Lincs.



There is a narrow gray ribbon cable (approximately 6“ long) that connects the power base to the first Linc on your stack (into the bottom of two connectors on the right side of the Linc when you are looking at the back). If you have additional Lincs, very short ribbon cables (approximately 1” long) connect each Linc to the one above it (from the top connector on the lower Linc into the bottom connector of the next higher Linc). These cables must be connected in the proper orientation – make sure the side of the cable with a small plastic projection on it goes into the connection with a corresponding small slot. Finally the power base lid should be installed on the top of the Linc stack. This lid holds everything securely in place and also completes a circuit that allows power to flow to the Lincs. This is a safety measure so you cannot get an electrical shock if you touch the inside of a Linc, which is exposed when the lid is not in place.

Each Linc has a user-modifiable “Linc Number” switch on the front. This is the “address” the PCI card (and therefore the software) uses when communicating with each Linc. The only absolute requirement is that each Linc in the stack have a unique number (from 0 to 7). However, we recommend that you use “0” for the top Linc and work down from there in sequence. If you are configuring two or four Lincs as a single station (read the next section), this is important.

After you have set the Linc numbers and connected all of the cables, turn on the power base power switch on the back of the power base and also turn on all of the Linc power switches on the right side of each Linc.

A Linc or half of a Linc (or maybe two or even four Lincs)

Before you ordered your system, you (or we) decided how many Lincs you need based on the number of stations you will be running and the response and stimulus requirements of each station. Each half of a Linc can receive input from four response switches and can control a number of stimuli (e.g., two feeders with magazine lights, two triple-cue devices, one tone, one house light, two “spare” devices, and two “auxiliary” devices). The two halves of the front of the Linc are labeled “A” and “B”. There are small lights that are lit whenever a response is sensed or a stimulus is operated. When you work with Graphic State Notation, you will see that part of the protocol creation screen includes a graphical representation of the available stimuli.

In addition, the front of the Linc includes three lights just to the right of the Linc Number dial which tell you whether the unit is sensing 28 volt power (the yellow light), 5 volt power (the red light) and communication signals from the computer (the green light). If all the cables are hooked up correctly, the computer is on, the power base is turned on and the Linc is turned on, all three of these lights should be lit.

Many (probably most) experimental environments require only one half of a Linc. If your environment has either more response switches or more output devices, however, you may need a full Linc for each station. For more complex environments (e.g., an 8-arm radial maze with photocell sensors at the entrance to each arm and also in the trough of each feeder), two Lincs may be required for each station. We also support a configuration in which *four* Lincs are used for each station. The Linc configuration which will be used for a particular project is specified when you set up a project database using Graphic State Notation, and it can be different for different projects. The details of how to specify the Linc configuration will be discussed later in a section entitled “Creating a project and a protocol.”

Input switches and output stimuli

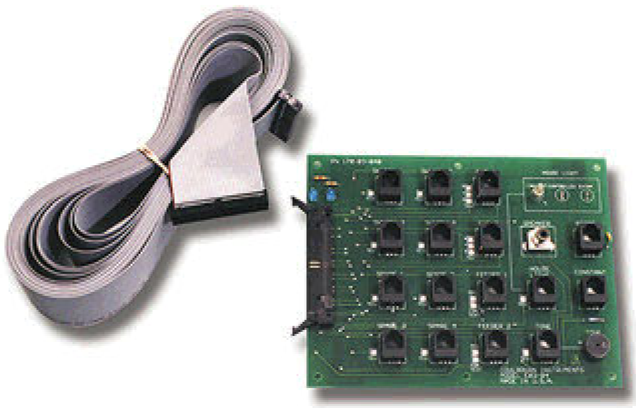
Every response that an animal can make using one of the Habitest modules causes the closing of a switch mechanism of some kind. That is why we refer informally to the devices, the connections on the ECB (described below), the logical names for the devices used in Graphic State Notation, and the responses made by the animals as “switches” or “switch inputs.” Examples of these devices include levers, photocell sensors (in lickometers, feeder troughs, nosepokes, arms of mazes, or elsewhere), bird pecking keys, contact sensors, ceiling chains, bump rods, activity monitors and hand-shaping switches (where the response is made by the experimenter rather than the animal).

The experimental environment presents stimuli to the animal. Examples of Habitest modules that present stimuli include house lights, ceiling mount bright lights, triple-cue lights, seven-pattern display lights, tones, shockers, pellet feeders, liquid dippers, liquid solenoid valves, grain feeders, olfactory stimulus controls, infusion pumps and guillotine doors.

There are other stimuli that may be arranged in the environment (such as cage floor type, hub shape, colored wall panels, the arrangement of hubs, cages and runways) that are not modified or controlled by the Lincs and the Graphic State Notation software. We are not directly concerned with these static stimuli here.

The Environment Connection Board (ECB)

The Environment Connection Board (or ECB) is where most of the Habitest modules in the environment (switches or stimuli) are connected to the Linc or Lincs for each station. All modules except those controlled by “auxiliary” outputs (e.g., shockers), have cables with telephone-jack-type connectors. These plug into appropriately labeled positions on the ECB.



Switch devices plug into one of the four connectors labeled “SWITCH 1”, “SWITCH 2”, etc. Triple-cue devices plug into “CUE 1” or “CUE 2”. A feeder (pellet feeder, dipper, grain feeder) plugs into “FEEDER 1” or “FEEDER 2”. Guillotine door lifters may be plugged into “SPARE” connectors, and so forth. If you have any doubts about what plugs in where, consult the H-Series User’s Guide.

Some devices include *both* a response switch and a stimulus. Examples of this include retractable levers, running wheels with a braking facility, lockable pass doors, and nosepokes. In these cases, there are connection cables for both the switch part and the stimulus part. There are other cases where two modules are hooked together and only one of them is connected to the ECB (e.g., the pellet feeders and the magazine light in the trough, tone generators and speakers, light intensity controllers and nosepokes). For the details, refer to the H-Series User’s Guide (also referred to as the Habitest Hardware Manual).

Each power base comes with one special module called a Probe. This is a device that looks like a pencil or pen but with the usual module telephone-jack-type connector. It is used for testing; plug it into the connector labeled “PROBE” on one of the ECB’s. When you touch the little silver area marked “TEST” next to any connector on the ECB, the system will react as if a response has been received from the environment or a stimulus has been turned on.

The gray ribbon cable that comes with the ECB goes from the connector on the side of the ECB (watch the orientation of the connector) to one of the wide connectors on the left side of the Linc (looking at the back). Either one or two ECB’s will be connected to each Linc. Unless you have an odd number of stations and you are using one half Linc for each station, there will be two ECB’s for each Linc. One will be for the “A” side of the Linc and the other will be for the “B” side.

Auxiliary outputs

Some devices are connected in a different way. Shockers are the prime example. These come with a pair of blue and black cables with mini-jack connectors; the cables go from the color coded connectors on the shockers to the same color coded “AUX” connectors on the back of a Linc. These connections are used to turn the shocker on and off from within Graphic State Notation. If you have a programmable shocker (the H13-16), a third cable will connect the shocker to one of the “PROG” connectors on the back of a Linc. This will allow you to set and change shock levels during an experimental run.

Creating your first database and your first protocol

Decisions you need to make at the beginning

By now you should have hooked up your hardware including connecting input and output devices to the Environment Control Board (ECB). That means you have decided which switch inputs are associated with which devices (levers, etc.) in the environment and which stimulus designators are associated with which stimuli (feeders, tones, triple cues, spares, etc.) You also know whether you will be using one half of a Linc, one whole Linc, or multiple Lincs for each station. When you create a database (also called a project), you will record some of this information for the program to use and also to help make your protocols more readable.

The decision about how much of a Linc (or how many Lincs) to dedicate to each station determines what the stimulus graphic looks like when you start to create a protocol. You will see on the screen the exact stimuli you have to work with. They have logical names (like “H LIGHT” for house light, “TONE” for a tone, “FEEDERS 1 and 2”) and as long as the devices have been connected to the ECB positions with the same names, your stimuli selections will be self-documenting. There may be complicated situations where you will be using these stimuli in non-standard ways – but not, hopefully, for your first project.

As you create your project/database, you will have the opportunity to give your own names to the input switches. It is not *necessary* to do so, but it will make your protocols more readable. Isn’t it easier to understand

IF 5 1-lever GO TO …

than

IF 5 1- GO TO …

Both statements are testing for five responses on switch 1, but the first way lets you know more about the way the environment is configured. For a *particular* protocol in this project, you might even want to provide a protocol-specific name for this switch

IF 5 1-left lever GO TO …

but we'll get to protocol creation a bit later.

There is one more decision to make before you create your first database. How often do you want the system to sample the environment to determine if the subject animal has made a response? You have three choices: once every 20 ms (50 times a second), once every 50 ms (20 times a second) or once every 100 ms (10 times a second). For most situations, 100 ms is a good choice. Most responses (e.g. licking or lever pressing) cannot be made faster than 10 times a second.

Before you run your first experiment collecting real data, you will want to design the *format* of your animal identifiers. You have a six-character field available to identify each animal. While you may use any numbers or letters that you wish, later retrieval and analysis of data will be easier if you make a plan before you start. You will have the option later of retrieving run data with animal ID's that match values in particular positions of this six-character ID. For example, you might want to reserve the first three characters to specify a group (e.g. C01, C02, D01, D02 for two control groups and two drug groups) and the last three for the animal number within the group (001, 002, etc.). Then you could ask for all data (for a given protocol, between selected dates) for a particular animal, for a particular group, for both drug groups combined, for the entire study, etc. The design of the animal ID is up to you, but remember that the position of each character is important – "D02001" is different from "D021 ".

Things you can add or change later

The decisions you have made above will only affect the *first* database that you are about to create. You may configure the switches and other devices differently for each database (although you won't want to do that without some good reason) and you may select different sampling rates for each database. You can also use different animal ID conventions at any time, although you will probably want to be consistent.

After you create a project database, you will write one or more experimental protocols that will control the logic of experiments. You can add new protocols at any time (up to 99 of them for each database), you can copy a protocol to start a new one, and you can modify any protocol that you have not yet run to collect "real" data. We'll go through a complete example in the next section of this tutorial.

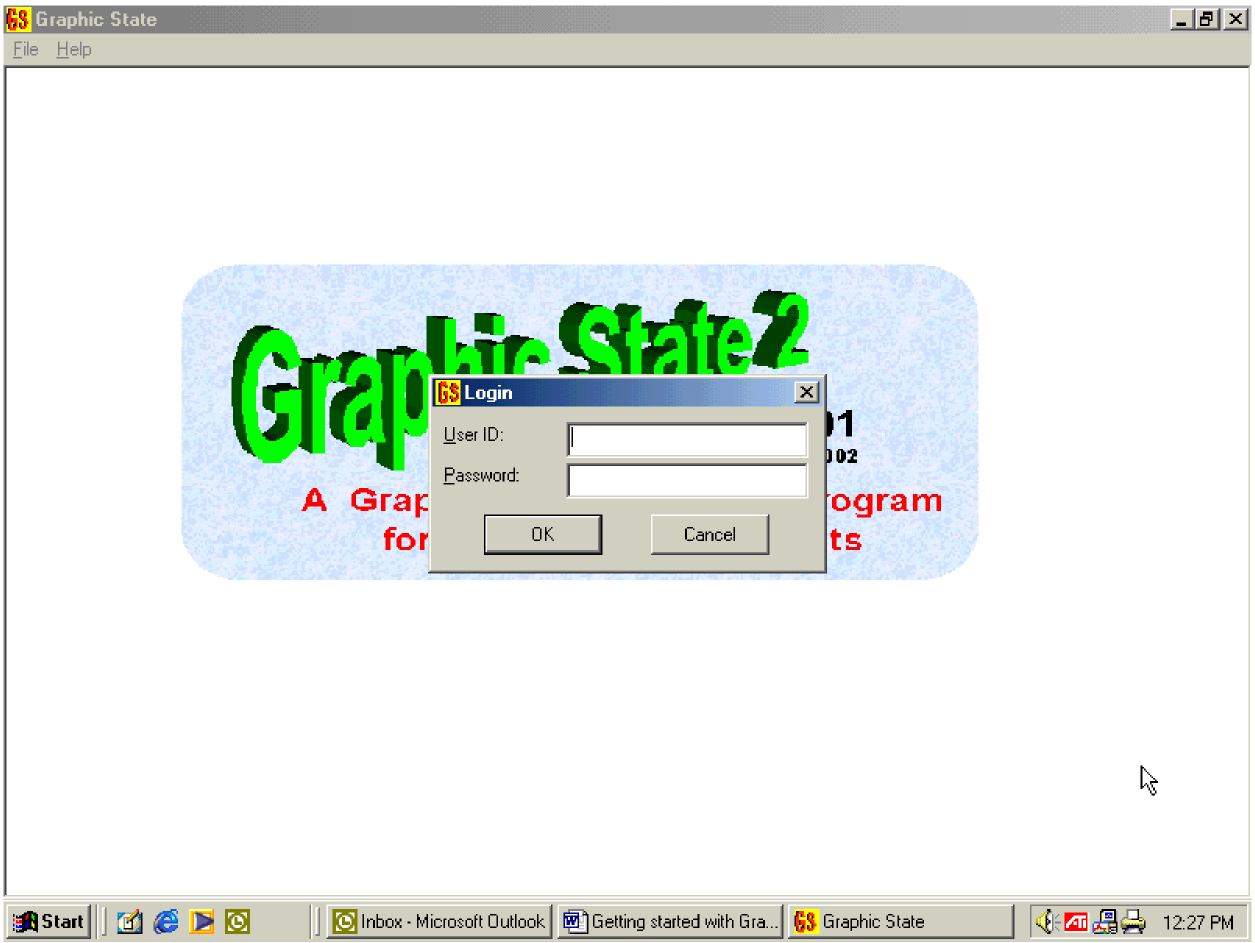
There may be repetitive kinds of information, useful in defining or running protocols, which you will want to store in lists. Some of these kinds of information may include ratio requirements for Variable Ratio schedules, time intervals for Variable Interval schedules, and the ID's of animals assigned to particular groups and particular studies. These lists can be created as the need for them arises. However, the *format* of animal ID's should be determined (as mentioned in the last section) before you run any real sessions.

A complete example

Lesson 1 – Creating a project and a protocol

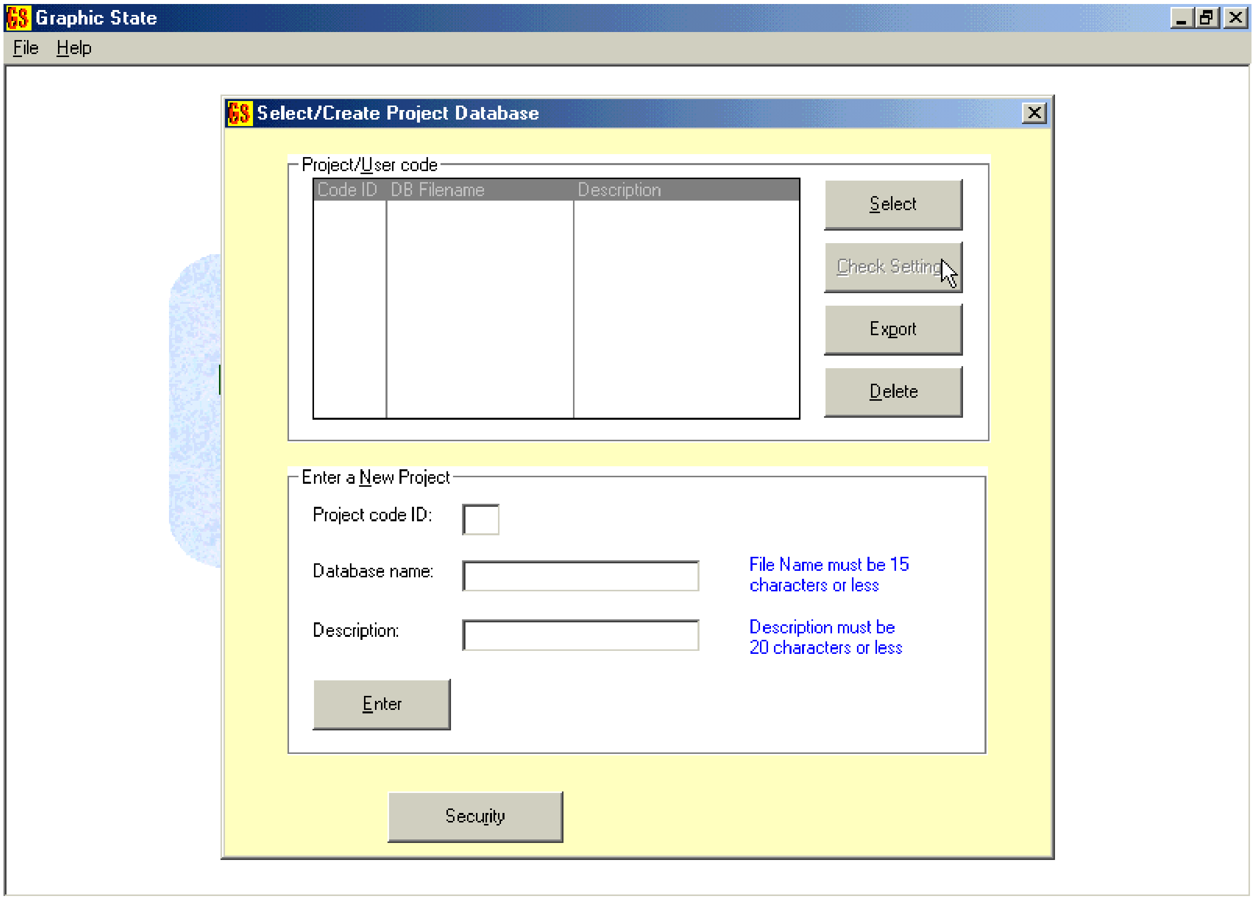
Finally we have come to the fun part – creating your first project and protocol. I am going to walk you through the creation of a basic CRF (continuous reinforcement) protocol where each press on a lever leads to the delivery of a food pellet. Then we will modify this protocol to first make it an FR5 schedule and then a VR5 schedule. More complicated (and perhaps more interesting) examples are in the last chapter of this tutorial and also available for download from our web site (www.coulbourn.com).

To begin, click on the Graphic State Notation icon on your desktop. This will bring up the Login screen. The system is shipped with Security enabled; the initial User ID is ADMIN and the password is PASSWORD.

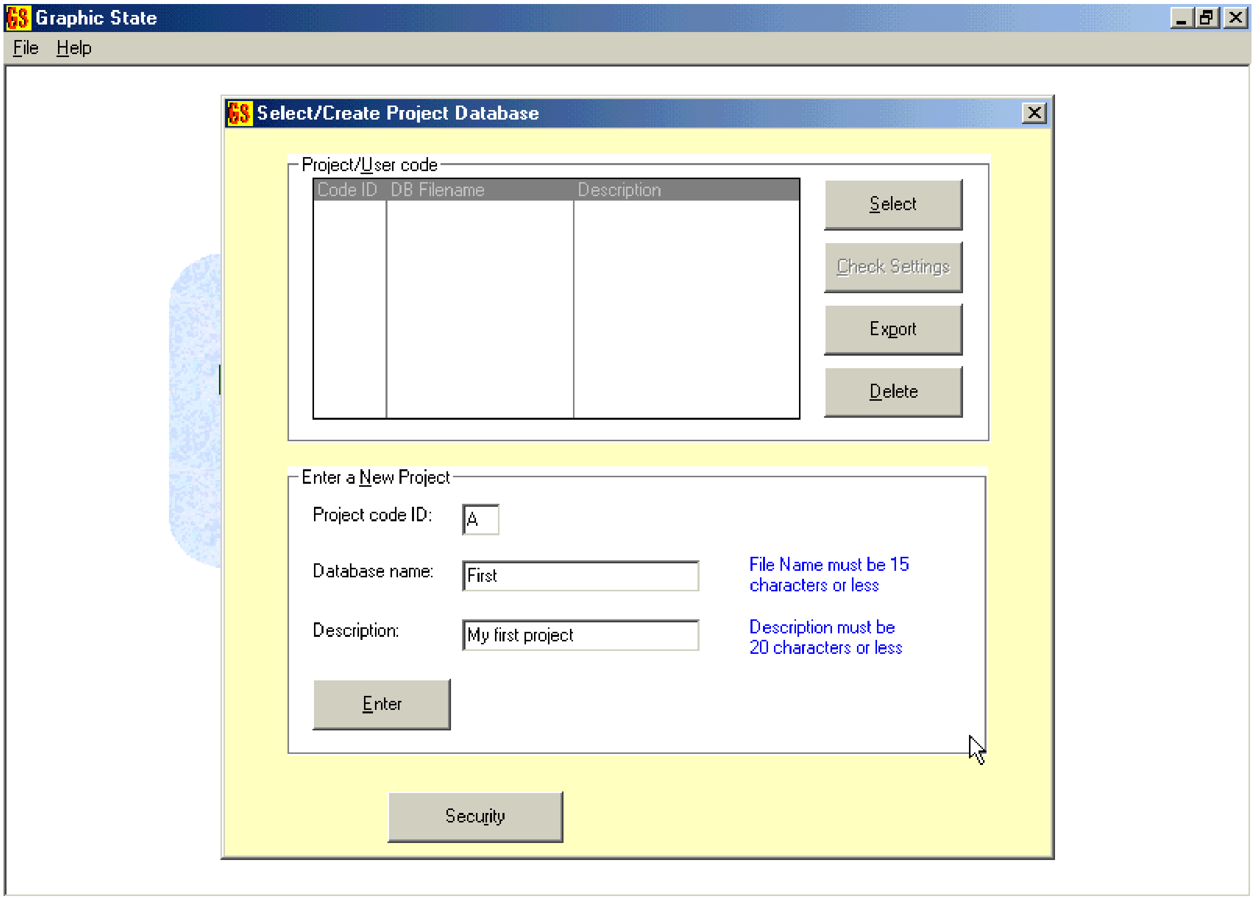


Enter these values and click on the OK button. Later you can turn the Security feature off or, better yet, set up a matrix of users and passwords that specifies what each user is allowed to do.

Next you will see the directory of existing Graphic State projects, of which there are none initially.

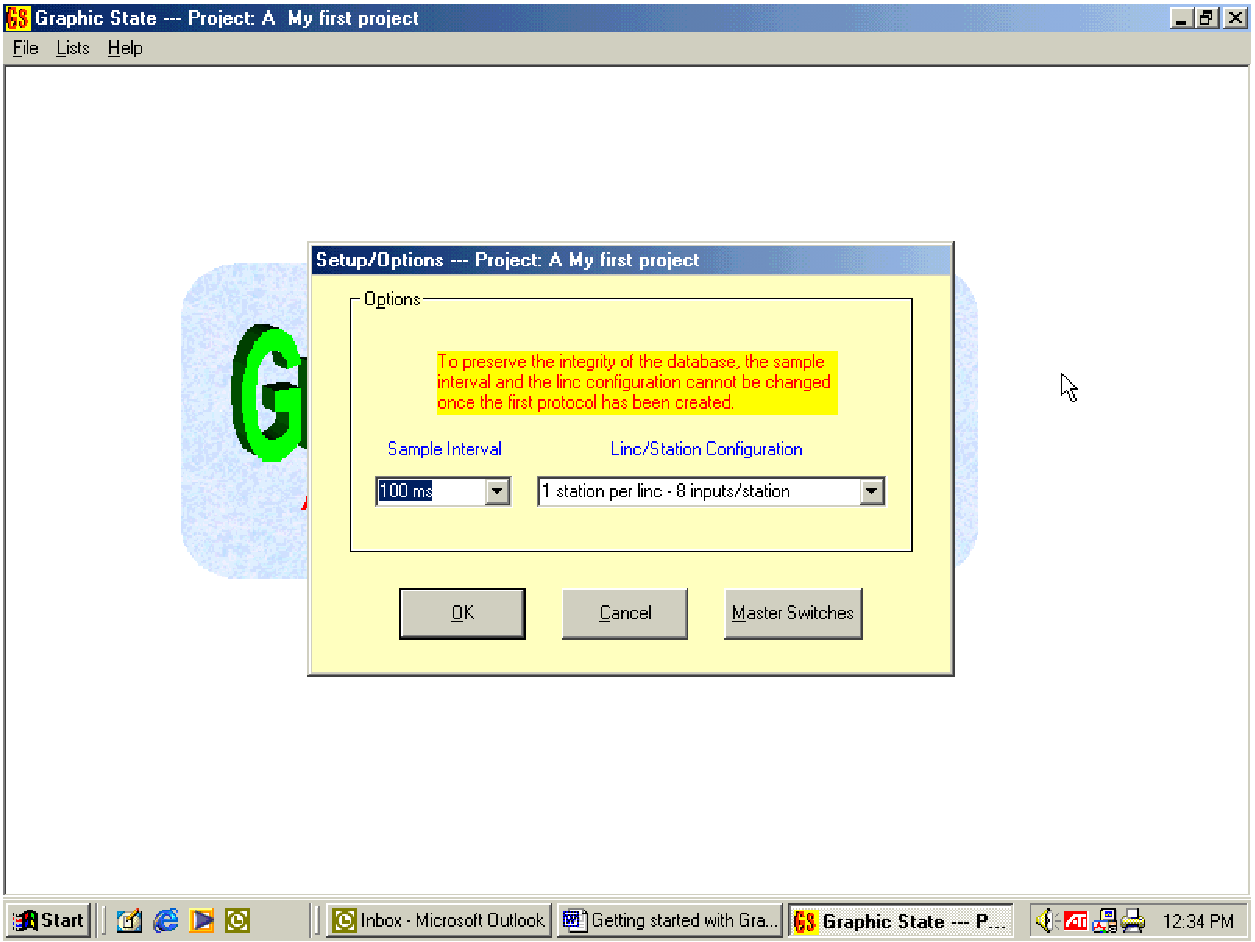


Place the cursor in the Project code ID field and enter the letter "A". Tab to the Database name field and enter "First". This will name your first project "A_First"; the file that will be created in the Datafiles directory (if you go looking with Windows Explorer) will be named "A_First.mdb". Tab to the Description field and enter any description you like.

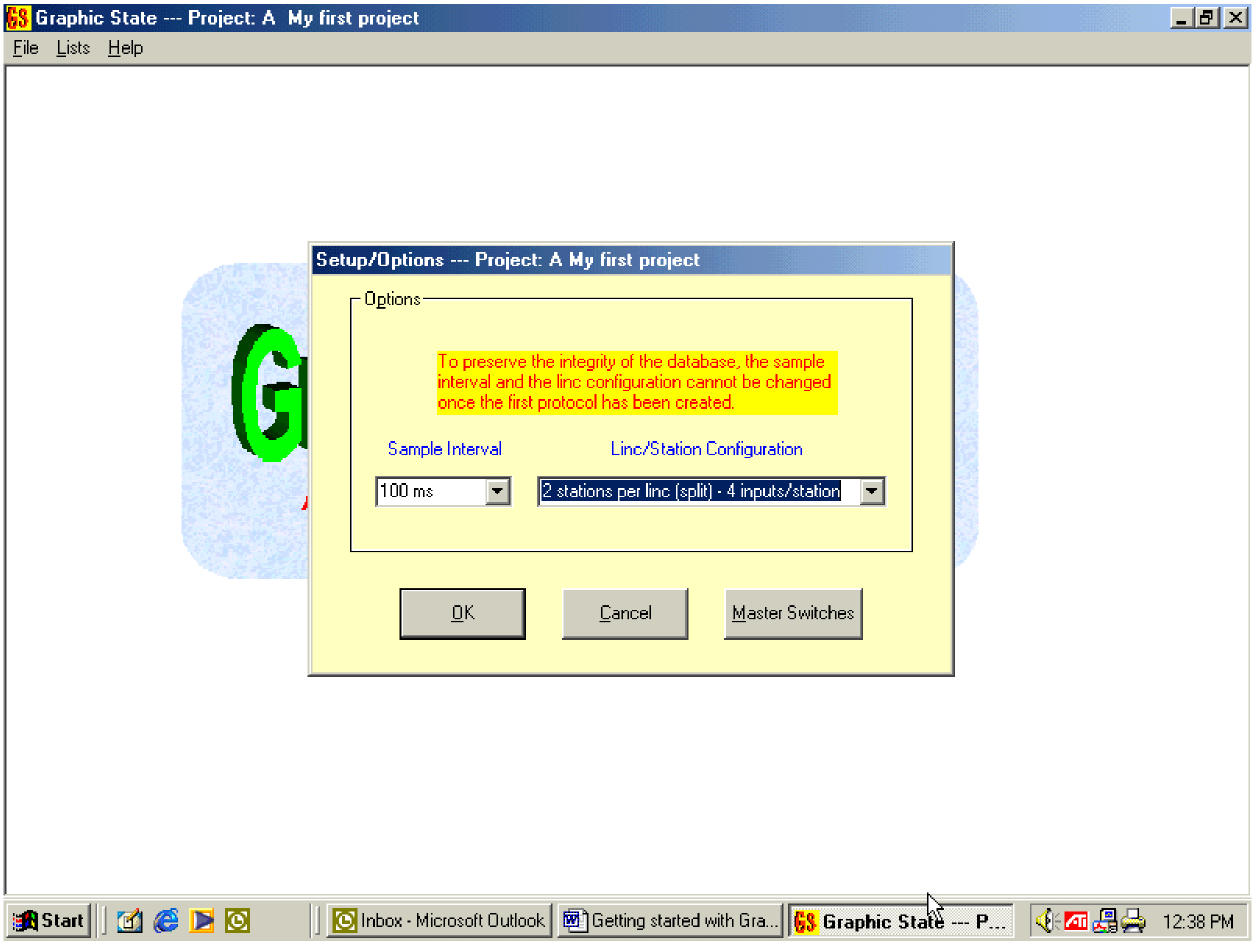


Click on the Enter key and move on to defining some of the database characteristics.

Default values for the sampling interval and the Linc configuration are shown.

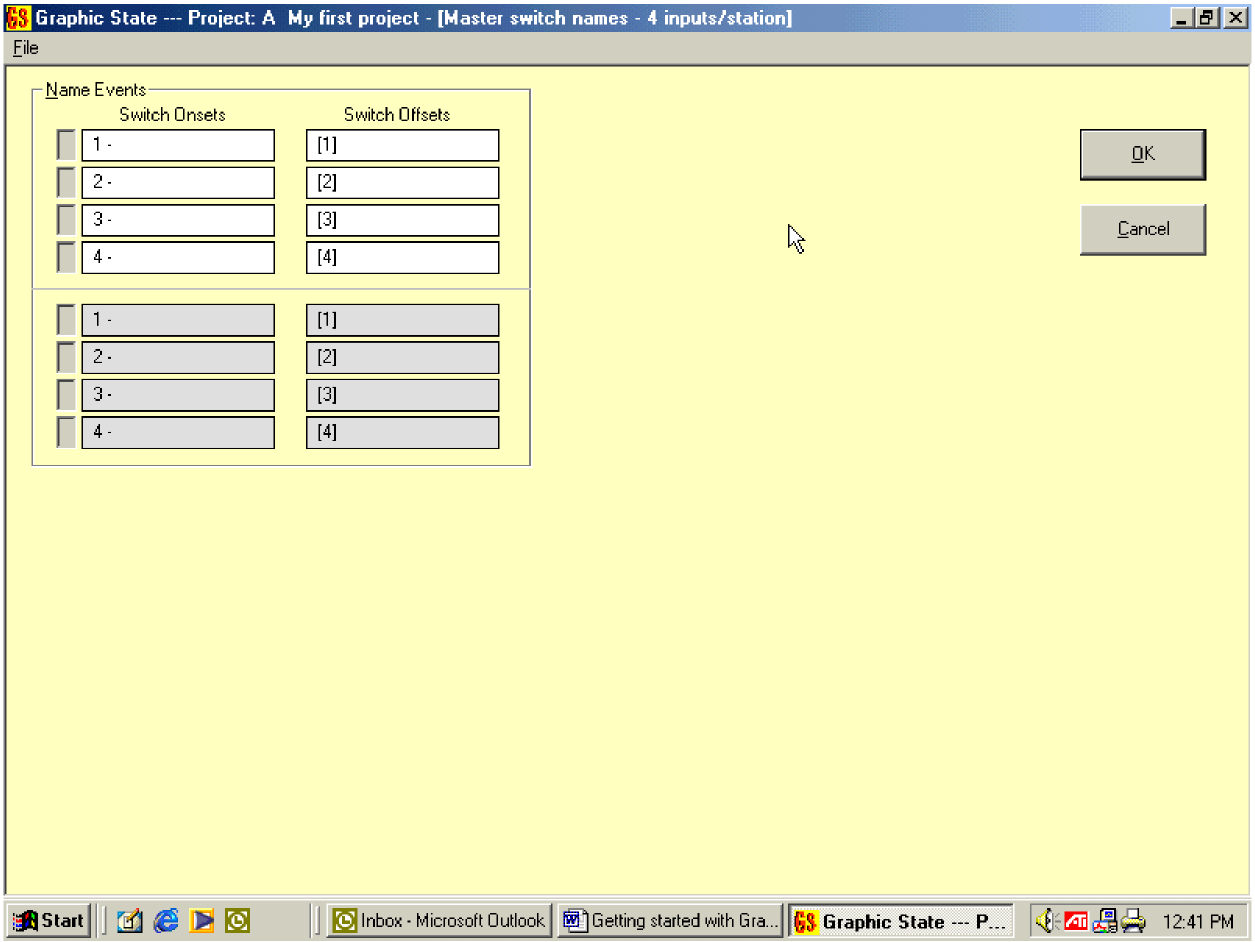


We will accept the default sample interval of 100 ms, but we want to change the Linc/Station configuration. From the drop-down list for that field, select the first option to specify 2 stations per Linc (i.e. one half of the Linc will be used for each station).



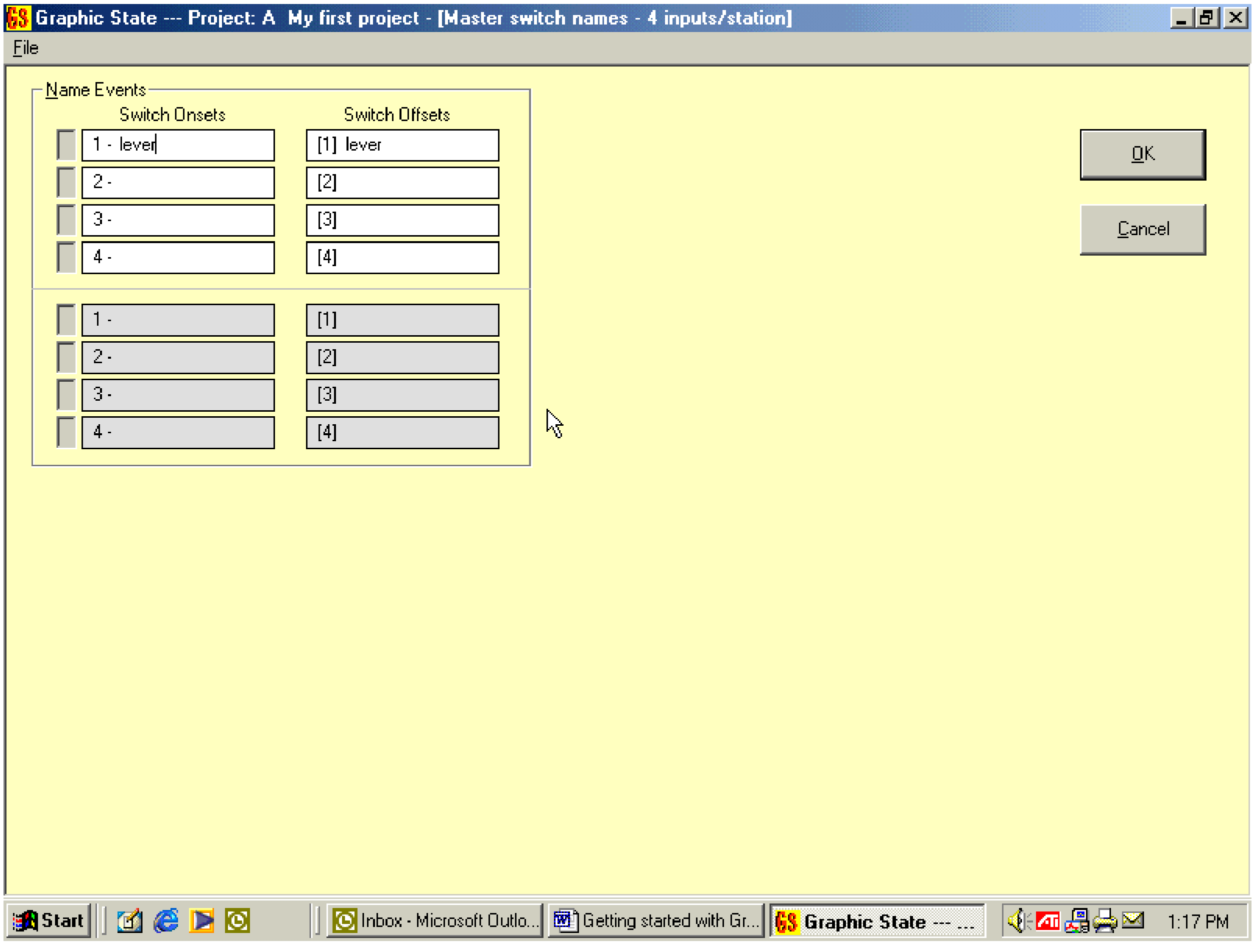
This is the most common Linc/Station configuration because most standard protocols can be implemented with four or fewer input switches, two feeders, one house light, one tone, etc.

Now click on the Master Switches key to enter names for the input switches.



Because we specified one half of a Linc per station, we have four switches that we can name. Actually, we can name both the *onset* of each switch and the *offset* of each switch. We can test either the onset or the offset or both in our protocols; this is one way to record the length of time that a lever is held down (or that an animal's nose stays in a nosepoke, for example).

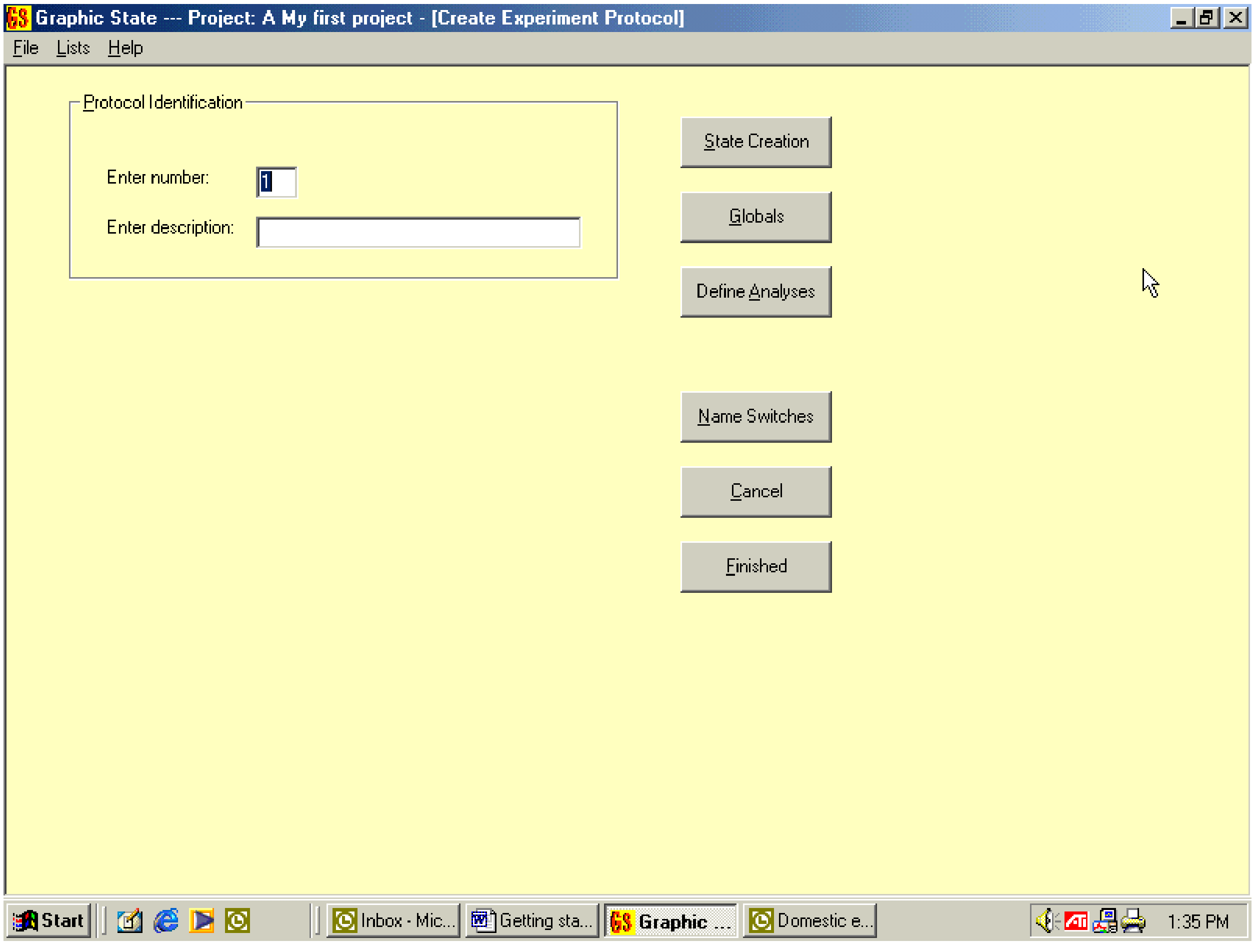
In our case, we are only interested in the onset of one switch – a lever. Position the cursor in the first Switch Onset field and type in "lever". The Switch Offset field is automatically given the same name, although we could then change it.



Hit the OK button to close this screen and then hit the OK button on the Options screen.

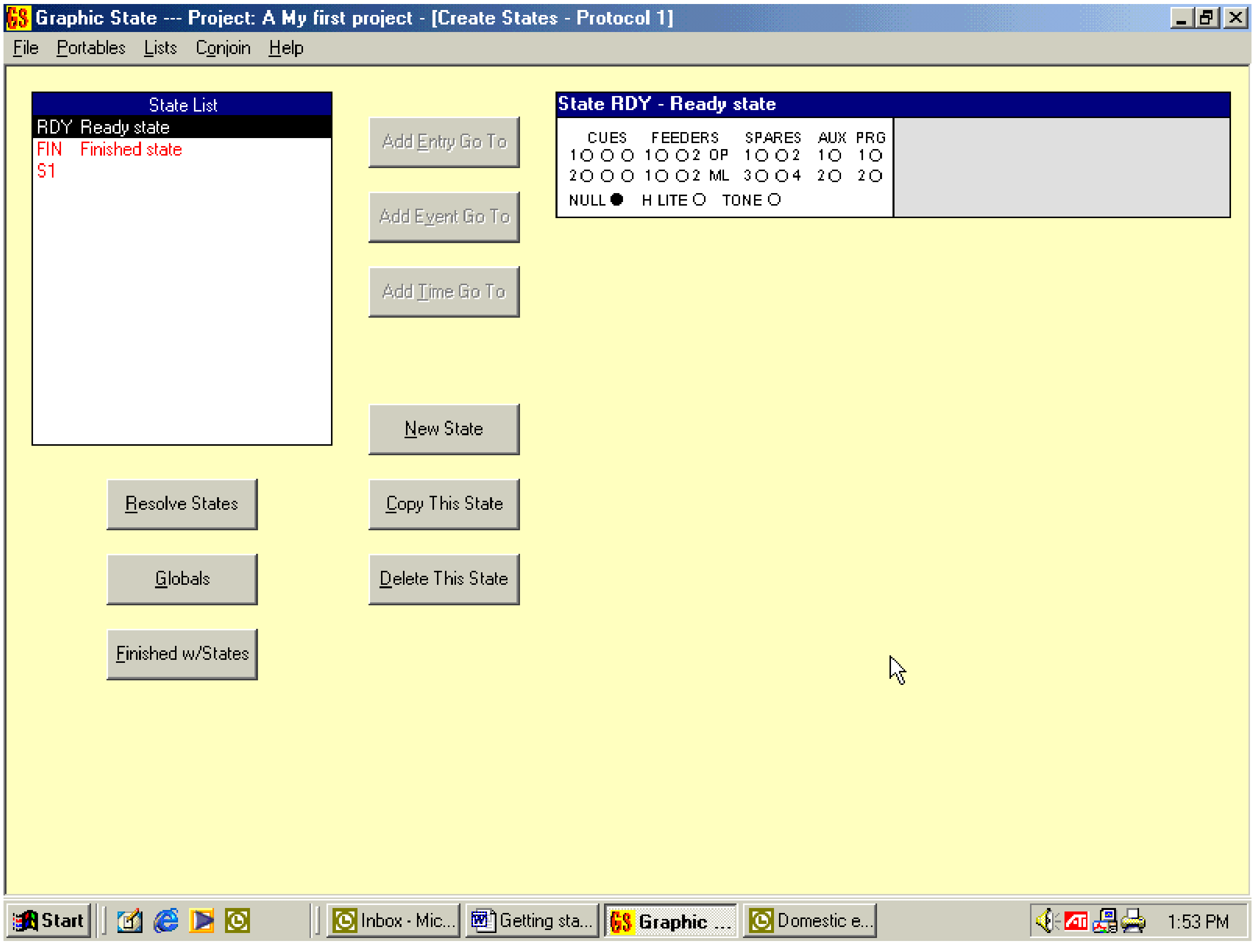
From the File drop-down menu, click on the first option Select Database. This will again display the index of all projects and our new one will be listed and highlighted. Click on the Select button to make it the active database. Actually, it already was the active database since we just made it, but if you exit the program and start again, you will need to go through these steps. Once you have more than one project, you can switch between them at any time by using Select Database, moving the highlighting cursor with the arrow keys, and clicking on Select.

Now, from the File drop-down menu, choose the Create Experiment Protocol option. This will bring up the first screen that is used to create a new protocol.



Each protocol is assigned a number. By default it will be assigned the lowest unused number (1, in this case), although you may give it a different unused number. Hit the tab key to move to the description field and type in a descriptive name for your protocol (how about “CRF”?).

From this point, you could create the states for your protocol, specify global stimuli or state flow conditions (that apply across all states), define ways to do data analysis (which we will save until later), specify protocol-specific switch names to be used instead of the database-wide switch names, or exit from the protocol creation process. Let’s start by clicking the State Creation button.



This is our starting template for creating a protocol.

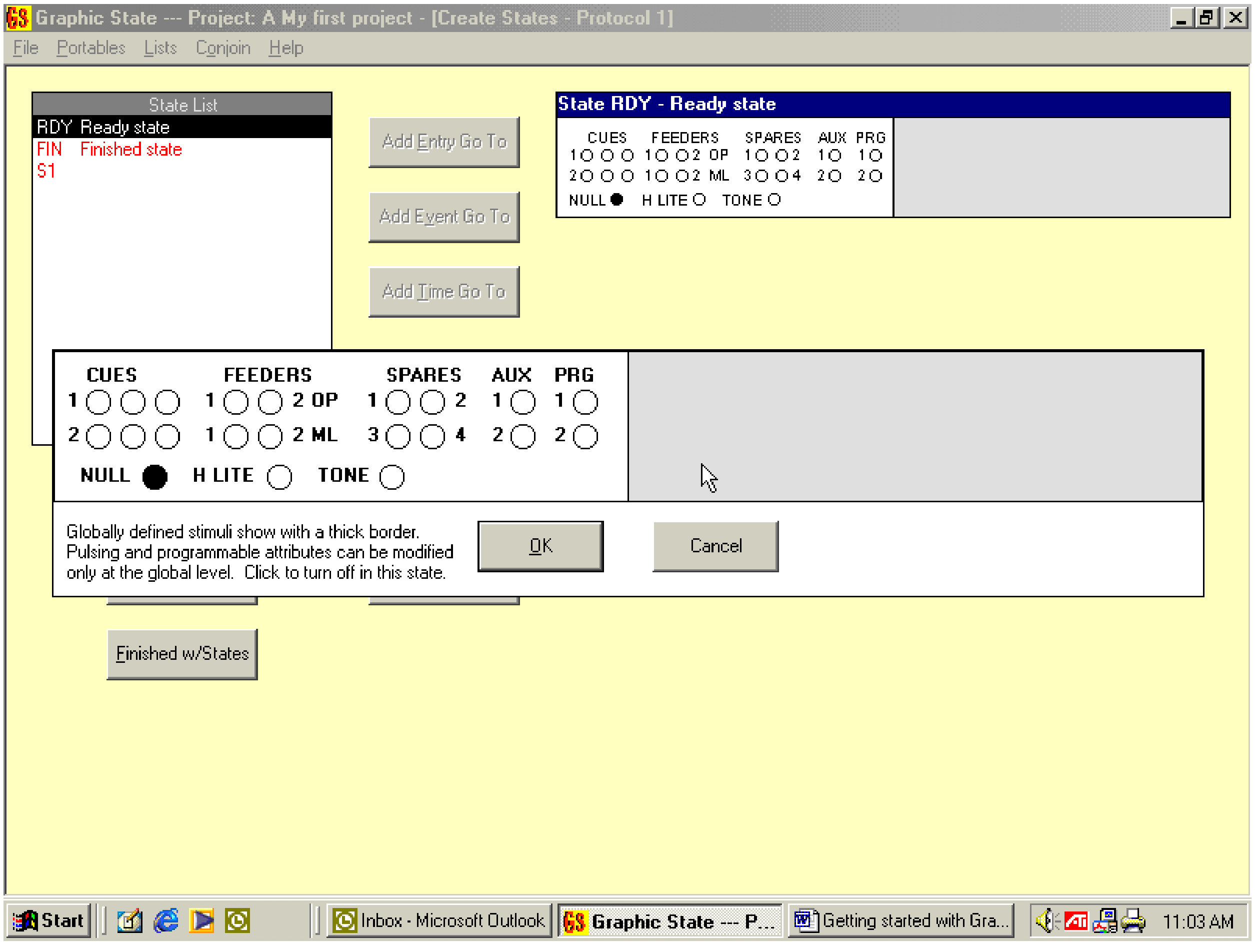
Three states automatically appear in the State List in the upper left. These are:

a. RDY – the “ready” state that will be active after you assign a protocol number and an animal ID to a station on the run screen but before you start the run (usually the animal will be in the station for part of this time)

b. FIN – the “final” state that will be active when the run terminates (again, the animal will be in the station at this time until it is removed by the experimenter)

c. S1 – the first state that will become active when the actual run starts

The RDY and FIN states may be left as they are (with no stimuli selected), or you may modify either or both of them to select stimuli that should be presented during them. For this example, let’s modify the RDY state to turn the house light on. Click on the RDY state line in the State List (if it is not already highlighted). A graphic display of all the available stimuli appears at the top of the right hand side of the screen. Double click anywhere in this graphic to produce a larger version of it that can be modified.

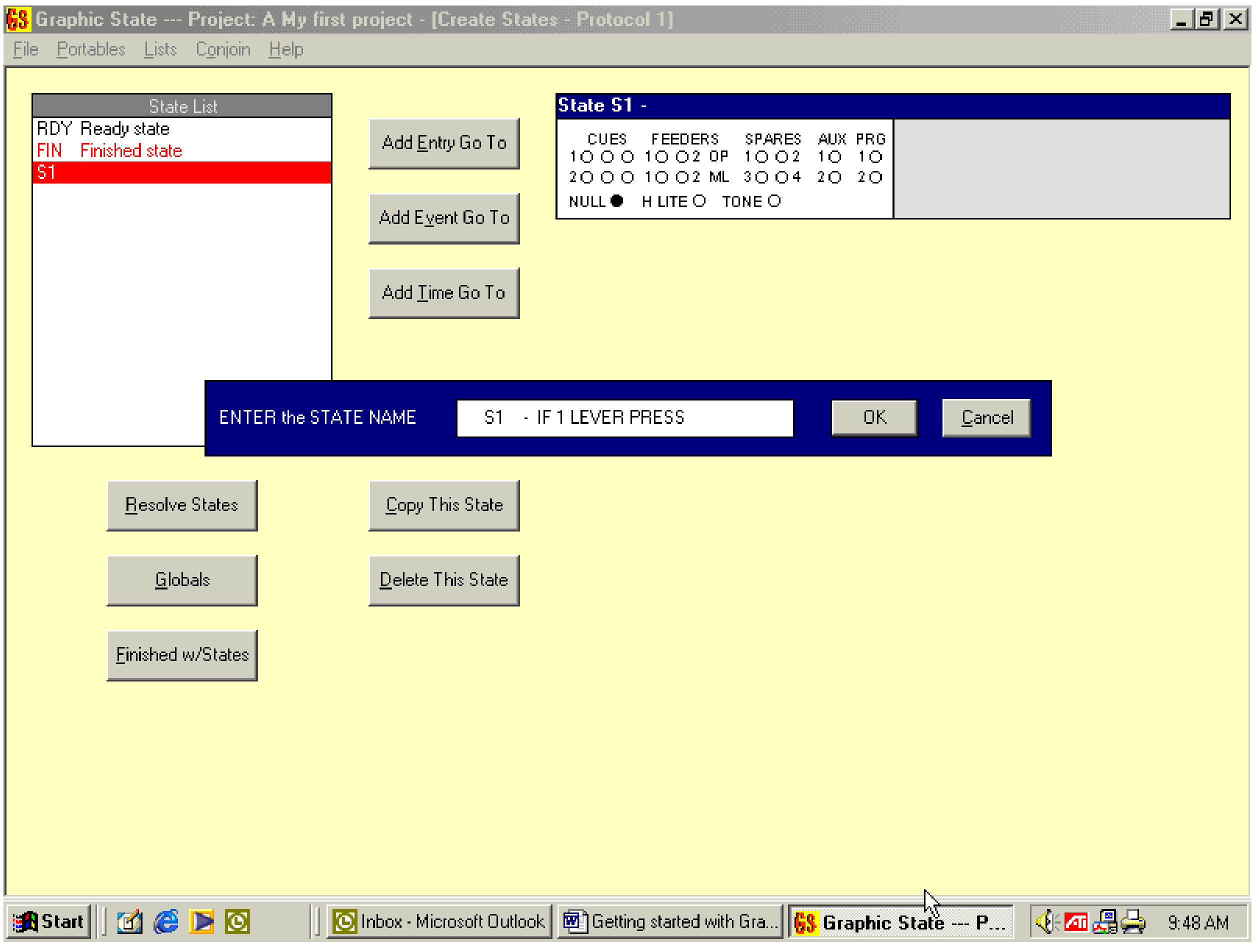


Click once on the button labeled “H LITE” and it will turn yellow. Click OK and the larger graphic will disappear leaving the house light button selected (yellow) in the smaller graphic. Now the house light (or whatever is plugged in to the house light connector on the ECB) will be on while the protocol is in the RDY state.

Select the FIN state from the State List and notice that the button named “NULL” is selected. This means that when the run ends, all stimuli (including the house light) will be turned off. When a particular stimulus is not selected in a state, it will be off in that state.

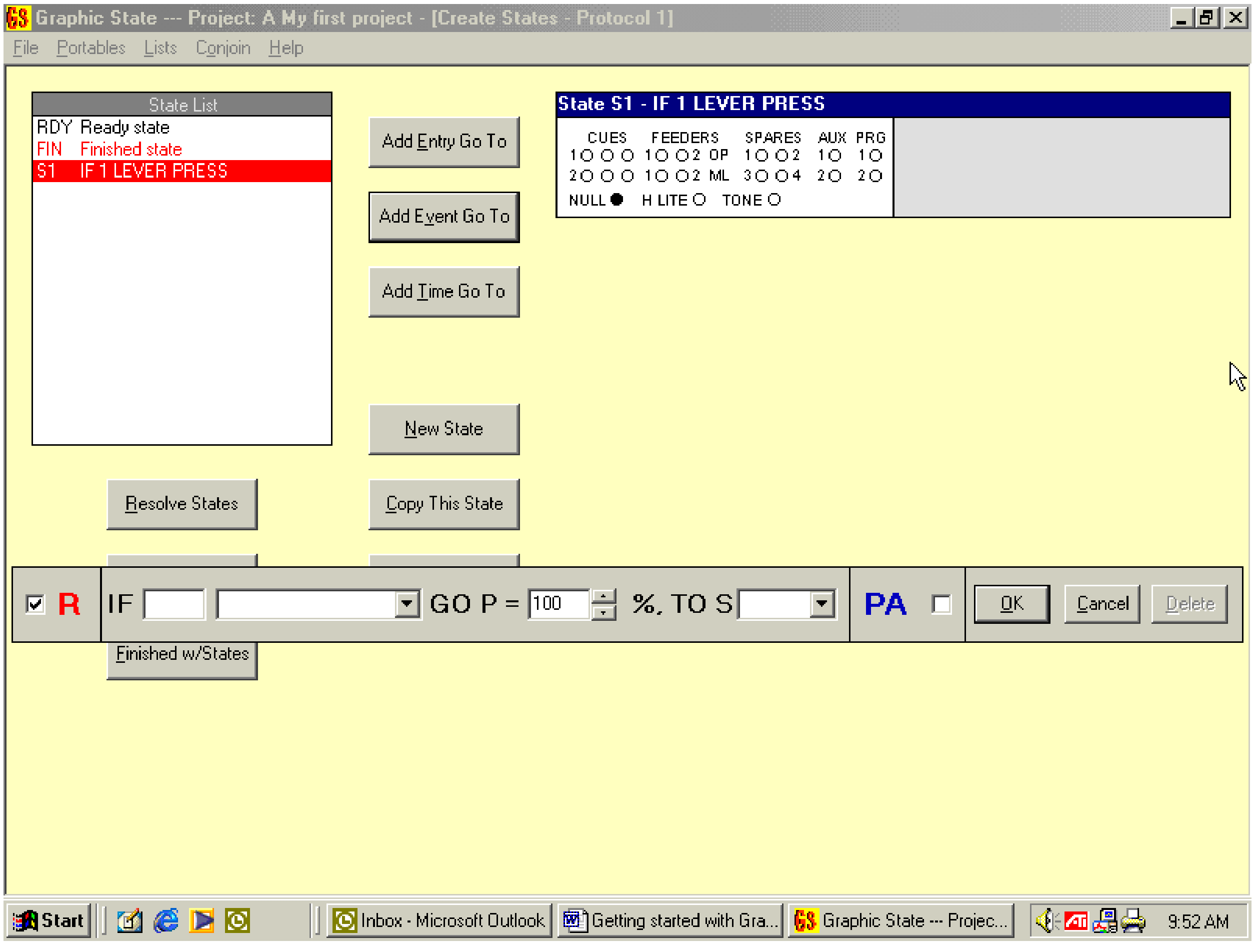
Now select the S1 state from the State List. This is where we will start specifying the logic of the protocol. We want to implement a CRF schedule. After every lever press, the feeder should be operated to deliver one pellet. In S1 we will test for 1 lever press response and when it has occurred, we will transfer control to another state (S2) where we will operate the feeder.

First, just for enhanced readability, let's give S1 a meaningful name. Double click with your mouse in the title area (the dark blue area) of the stimulus graphic. Type in the name or descriptive phrase that will explain what is happening in this state.

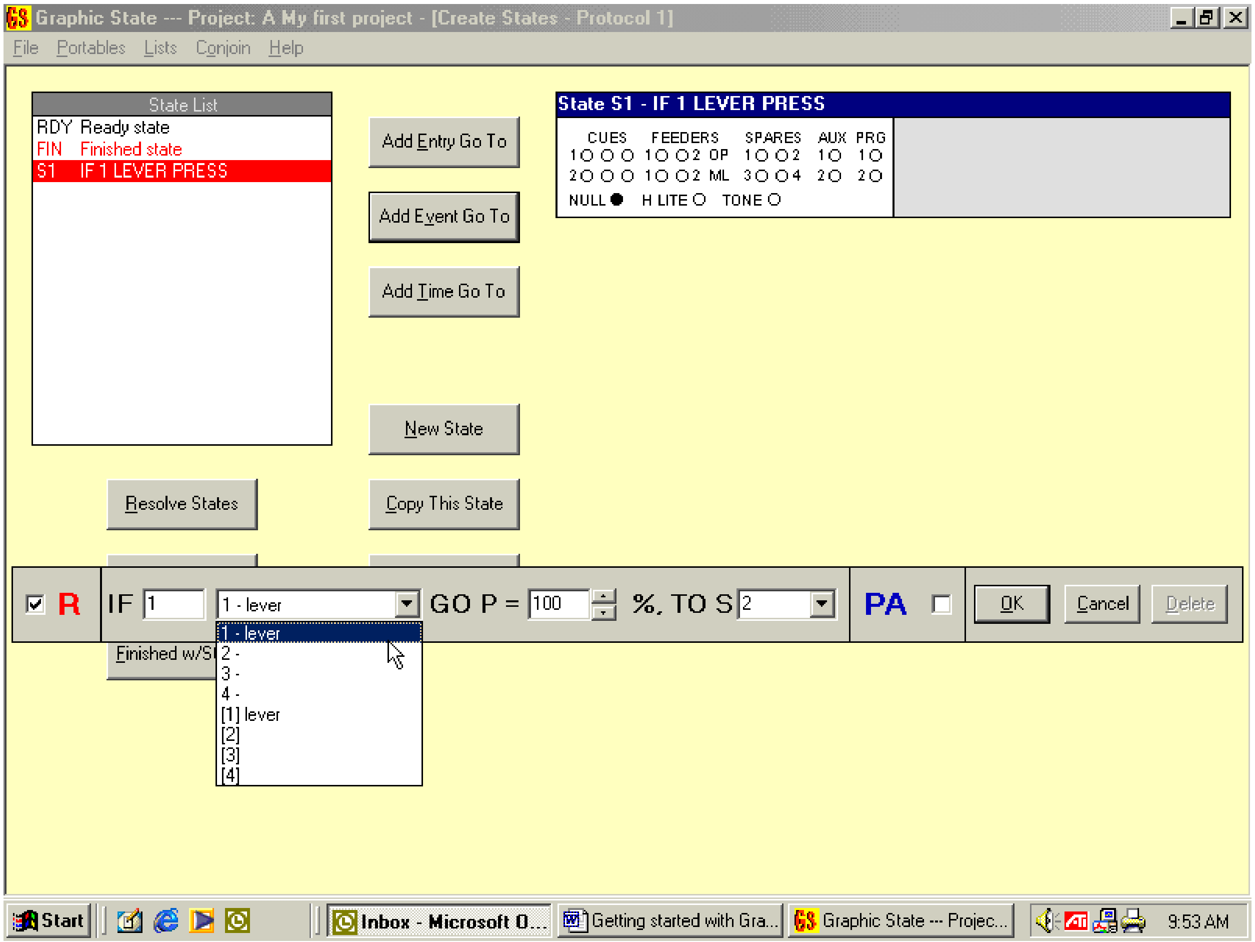


Click OK and the name will now appear both on the stimulus graphic and in the State List.

Now click on the "Add Event Go To" button. This will bring up a template for creating an IF statement to test for responses.

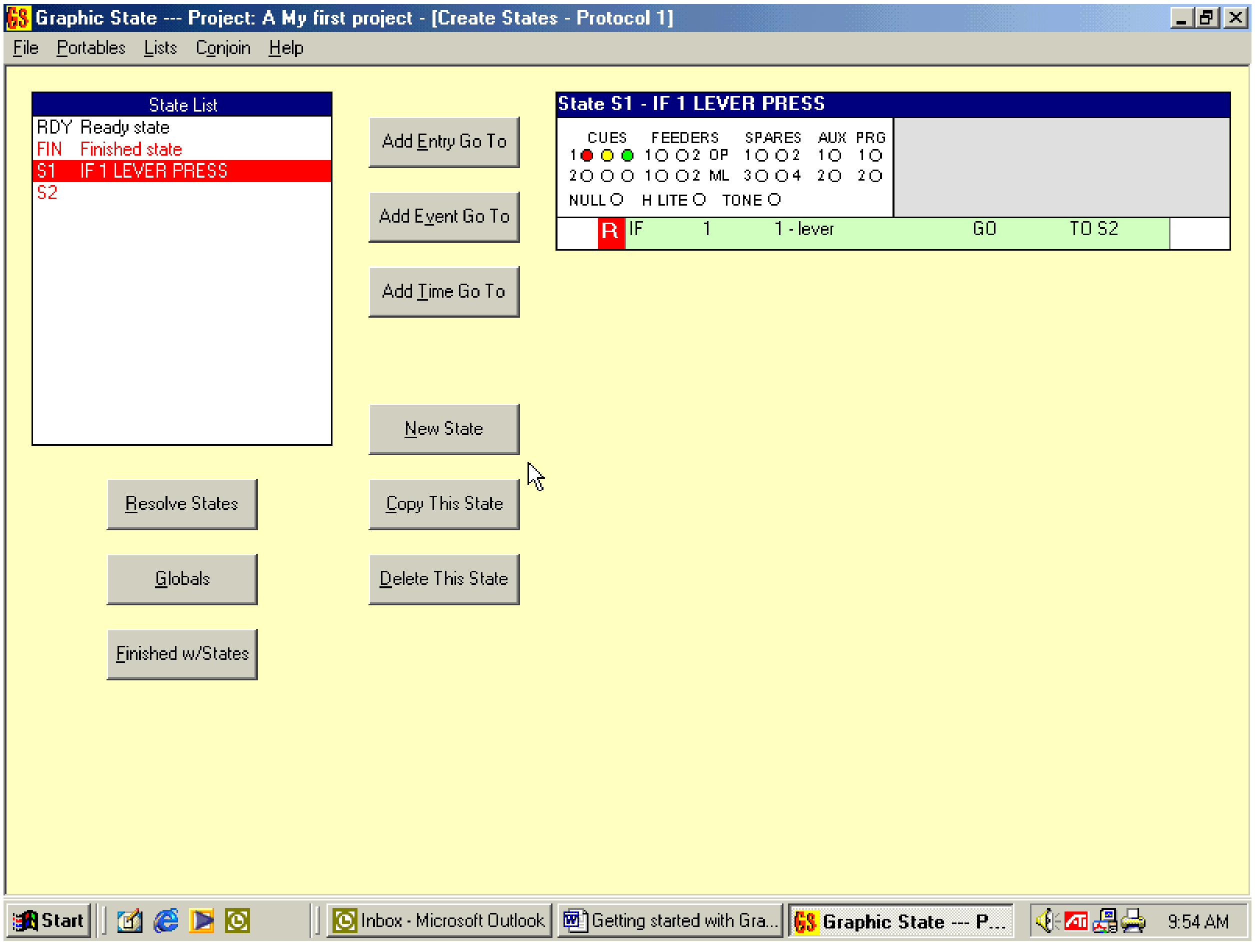


We'll go over more of the information here a little later, but for now start with *the* first white field following the IF where the response requirement (i.e. the number of responses) is entered. For the second field, you will select from a list of possible switch inputs (click on the down arrow to see the list). In the third empty white field you will enter the number of the state to which you want to transfer control. For this protocol, we want to specify that IF 1 lever press (switch 1) is recorded, control should GO TO S2.



Click OK to close this window and accept the test. It will now show up under the stimulus graphic.

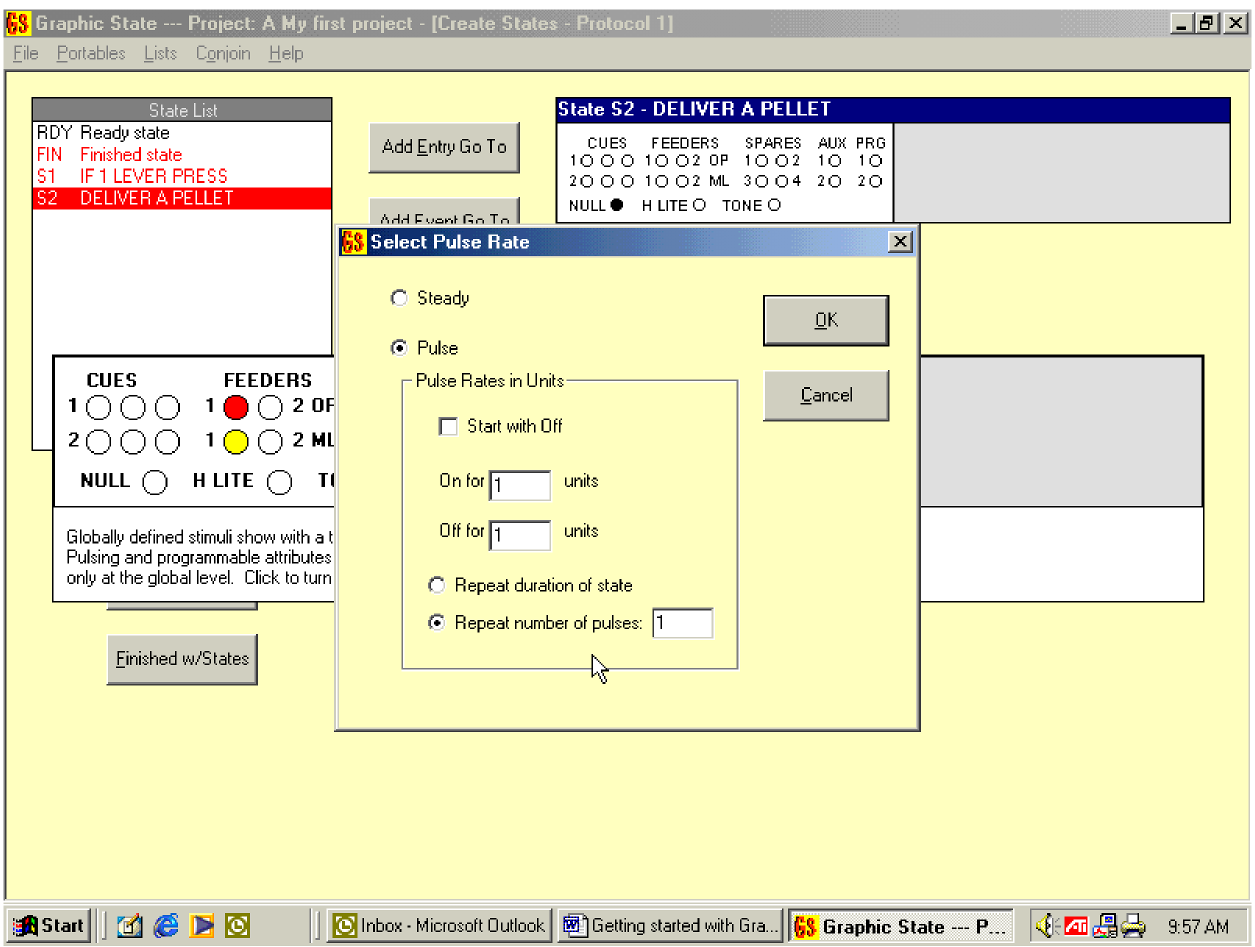
Speaking of the stimulus graphic, let's specify a stimulus to be displayed during this part of the protocol. Double click on the stimulus graphic to make it larger and available for changes. Click once on each of the three buttons grouped together as the first set of "CUES" and then click OK. You have now turned on all three lights in a triple cue. Your screen should now look like this:



Note that S2 now appears in the State List. When you name a new state (which you did in the GO TO from S1), it is automatically created.

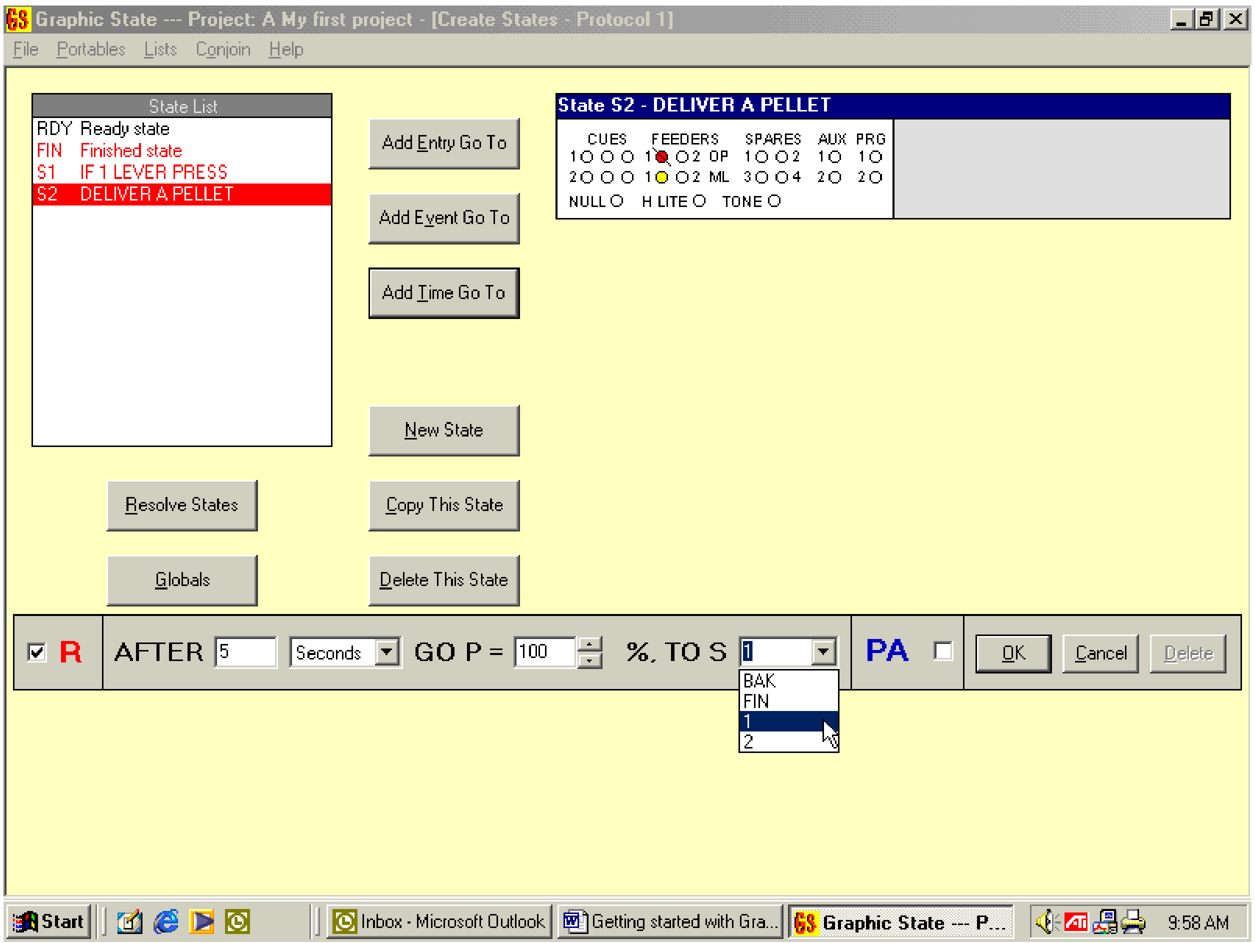
Click on the S2 line in the State List to select it. In this state we want to deliver a single pellet from Feeder 1, light the magazine light for a short period, and then return to S1. Double click on the blue area at the top of the stimulus graphic, enter an appropriate name for the state (e.g. “DELIVER A PELLET”), and click OK. Next, double click anywhere in the stimulus area to bring up the larger graphic. There are four buttons under the heading “FEEDERS”. The top two operate the two possible feeders and the bottom two turn on the magazine lights. The left buttons are for Feeder 1 and the right buttons are for Feeder 2. Click on the “OP” (for “operate”) and “ML” (for “magazine light”) buttons for Feeder 1. You should see a solid red button and a solid yellow one.

We want to remain in S2 to turn on the magazine light for a noticeable period of time (e.g., 5 seconds). However, if we operate the feeder for this period of time, we will deliver MANY pellets. We want to operate the feeder for the shortest possible period of time (1 sampling unit) to deliver only a single pellet. We will do this by “flashing” the pellet feeder instead of leaving it on for the entire duration of S2. Double click on the red Feeder 1 “OP” button to bring up a menu that allows us to specify flashing parameters. Select “Pulse” instead of the default, “Steady”. Fill in the blanks to specify “On for” 1 units (and you must also specify an “Off for” time although the exact number is not relevant in this particular situation) and then select “Repeat number of pulses” and enter a 1.



Click OK and the screen will close. You will now see a *blinking* red “OP” button. This is a reminder that the stimulus will be pulsed. Click OK again and the large stimulus graphic will disappear. The red "OP" button in the smaller graphic does not blink, but it has a line through it.

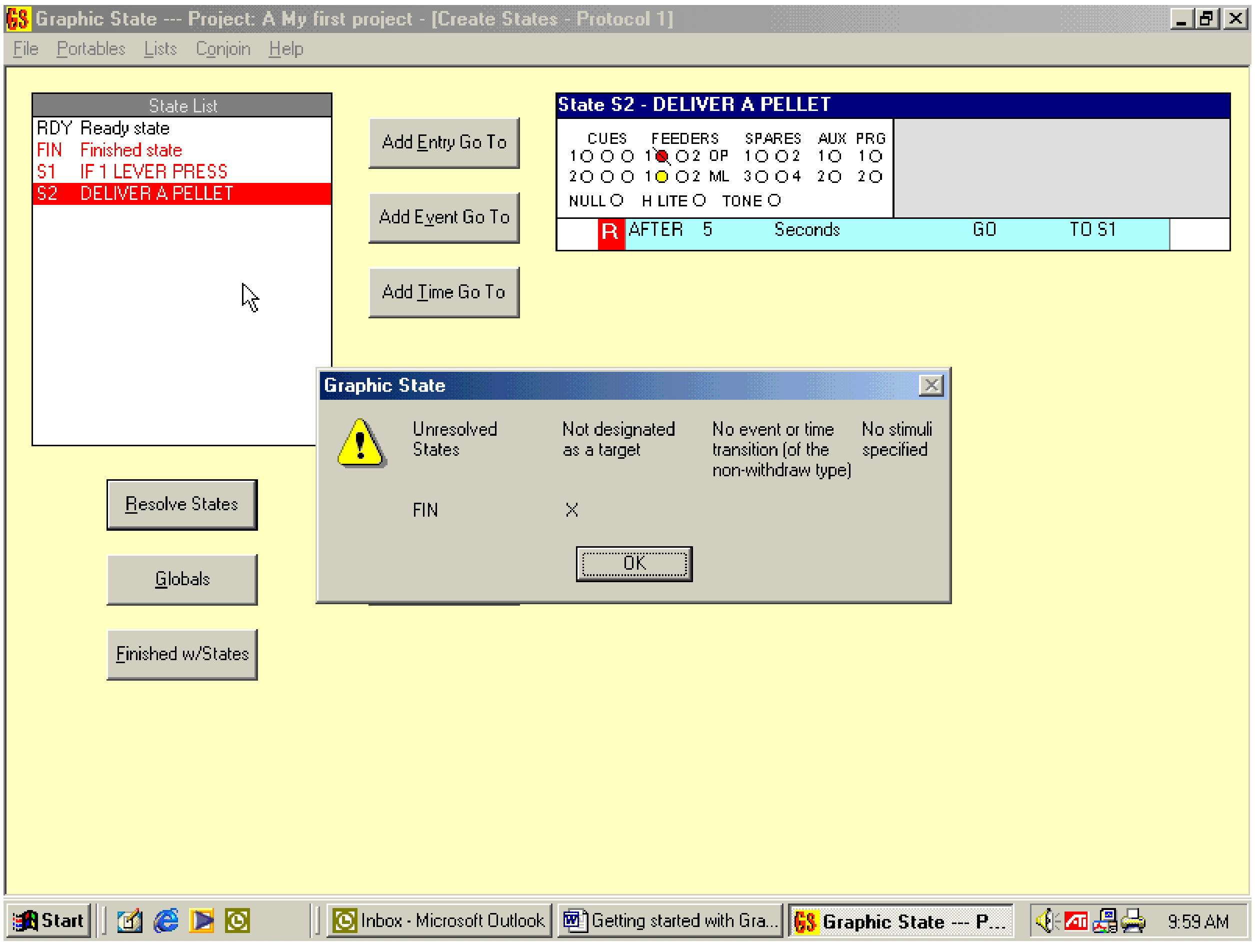
Click on the "Add Time Go To" button to enter the time parameter for leaving S2. Fill in the blanks to specify that AFTER 5 seconds, control should GO TO S1. Notice that there is a pull down list for time units and another one for previously defined states (although you could have typed in a new state here).



Click OK to finish adding the time statement to the protocol.

You now have a protocol that checks for lever press responses once every 100 ms (the sampling time unit) and when one has been detected, delivers a single pellet, operates the feeder magazine light for 5 seconds and then returns to check for additional lever press responses. The house light is on during the RDY state before the run begins. All three lights of a triple cue are on during the time that the protocol is checking for lever presses.

At this point, click on the "Resolve States" button to check the syntax of the protocol.

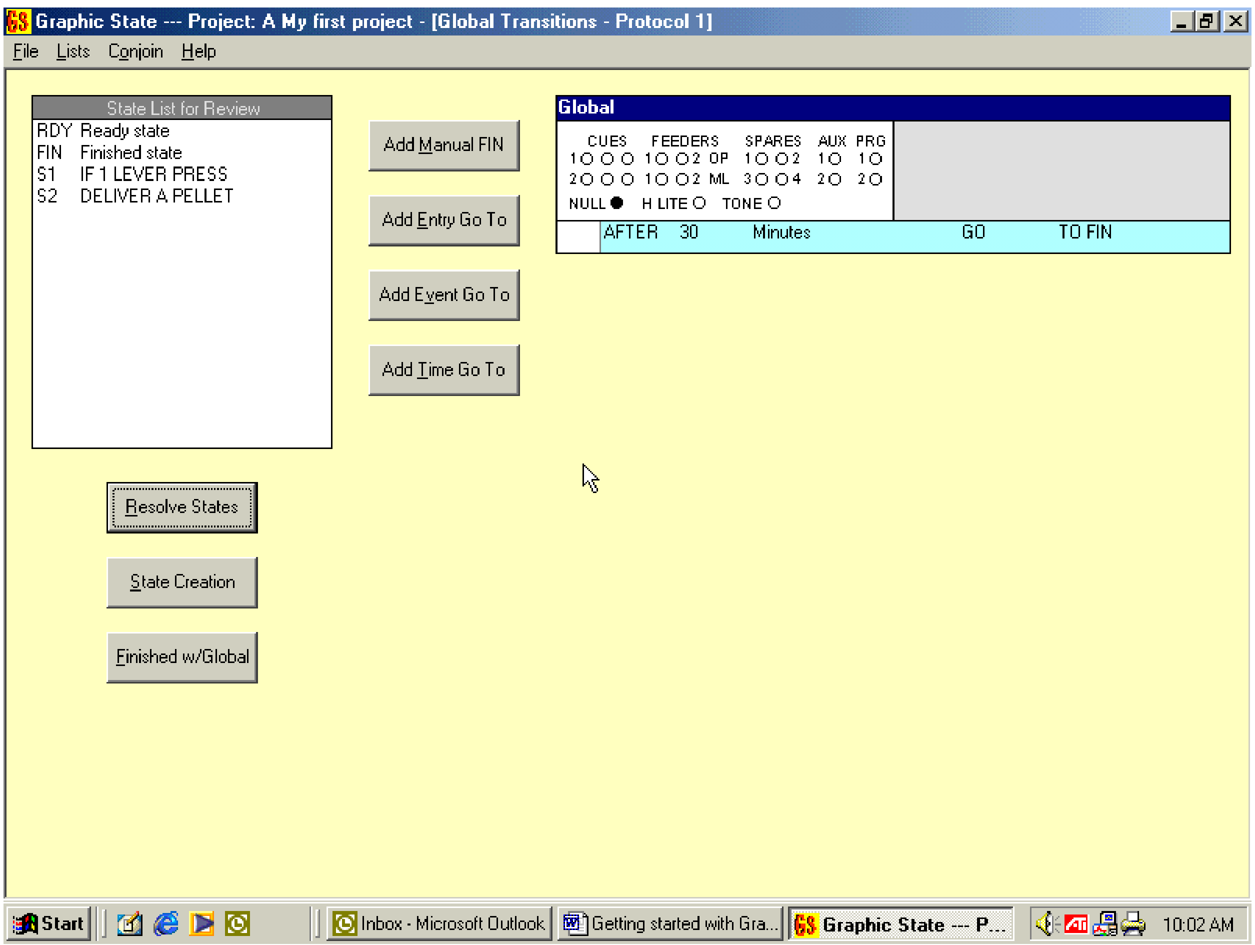


An error message comes up to tell us that the state FIN is not designated as a target. We have not specified the conditions under which the experiment run should end (i.e. GO TO FIN). Click OK and we will correct this. Notice that after you click on OK, the state descriptions in the State List for all of the states except FIN have turned from red to black. FIN is still red; this tells us that there is still something incomplete about this state.

There are MANY ways to end a run. For this example, we will use a global time parameter to end the run after 30 minutes. We have been working on the State

Creation screen. Now we want to switch to the Global screen; click on the "Globals" button to make the switch. This screen looks very similar (although there are a few differences like the absence of the buttons to create, copy and delete states), but the specifications we enter here apply across the entire run and not just to a specific state.

Click on the "Add Time Go To" button and enter the specification to GO TO the state named FIN after 30 minutes. Click on Resolve States again, and see that everything now checks out.

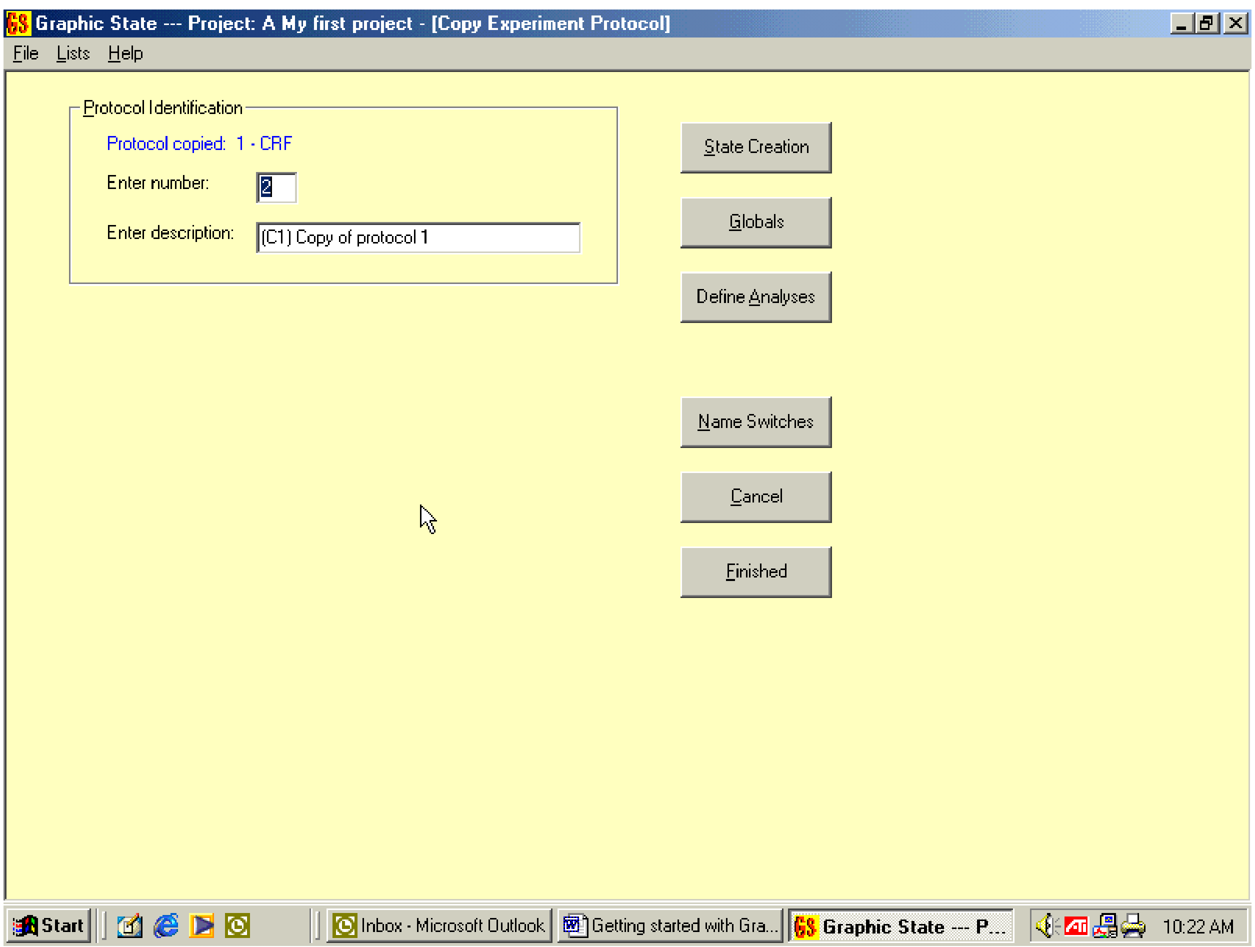


Click on "Finished w/Global" to return to the first screen for this protocol and then on "Finished" to return to the main menu screen. Select Exit from the pull-down list under File to terminate the Graphic State Notation program.

Lesson 2 – Copying a protocol to make a new one

Let's make a copy of our CRF protocol and change it into an FR5. Click on the Graphic State icon on your desktop to start up the program and enter your User ID and Password (ADMIN and PASSWORD if they have not been changed from the factory set values.) A list of all active Project Databases will appear; at this point there should be only one and it is the Project you created in Lesson 1. Highlight it (it should actually already be highlighted) and click on the Select button.

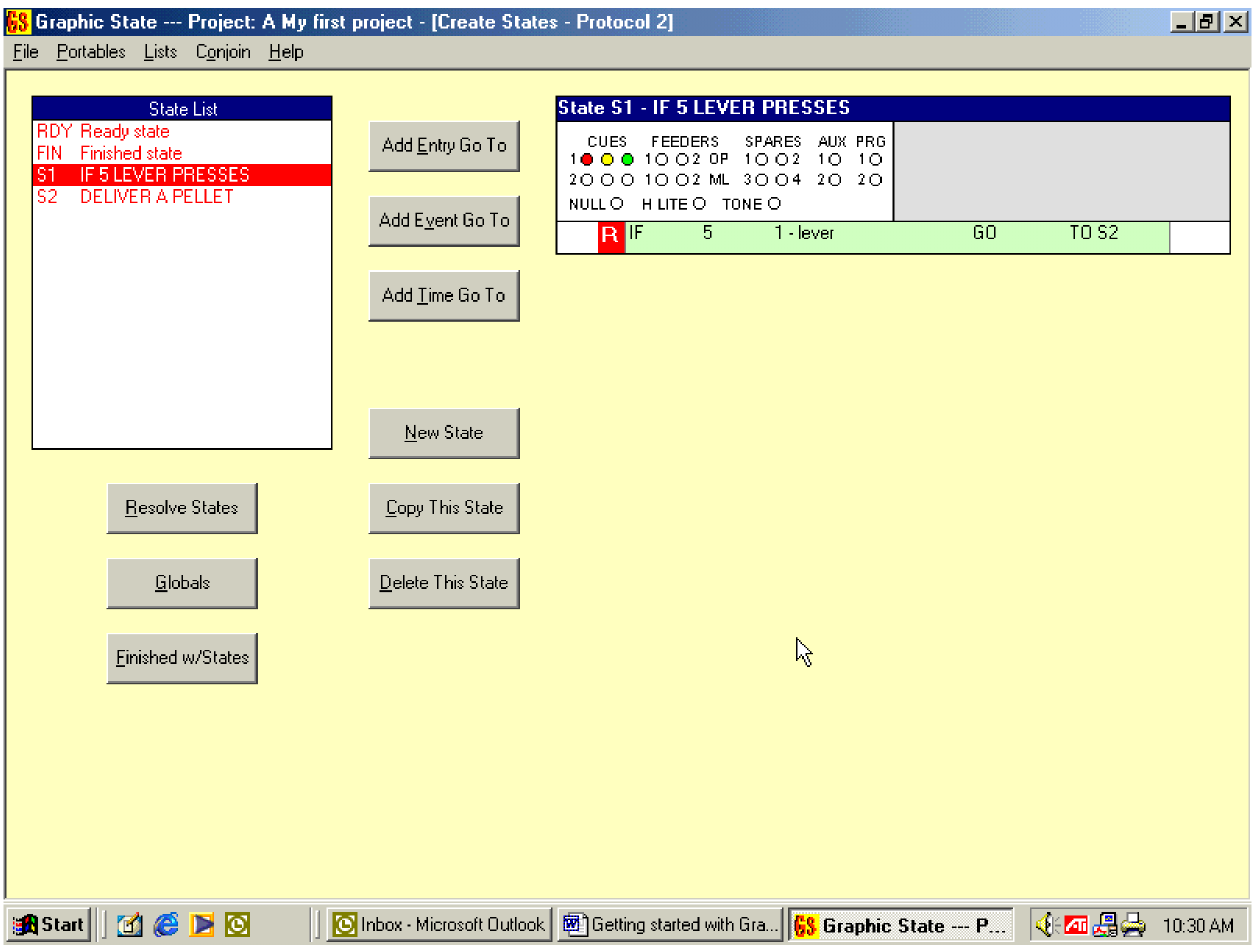
From the File drop-down menu, select Edit/Review Experiment Protocol. Protocol 1 is listed there in green with a status of "Ready to run". Click on the Copy button to immediately create a new protocol.



The new protocol was given the next available number (2) and a description with two parts. The first part "(C1)" is meant to leave a trail showing the origin of the protocol.

The second part "Copy of protocol 1" does the same thing, but we will now change that to be more descriptive. Click anywhere in the text of the description to highlight it and then type in the new description "FR5". Now click on the State Creation button.

We want to change the ratio requirement from 1 to 5. Select S1 and then double click on the green IF statement under the stimulus graphic. Change the "1" to a "5" and click OK. Also change the state name; double click in the blue area on the stimulus graphic and change the name to read "IF 5 LEVER PRESSES". This is not required for the protocol to run correctly, but it will be very confusing to leave it unchanged.

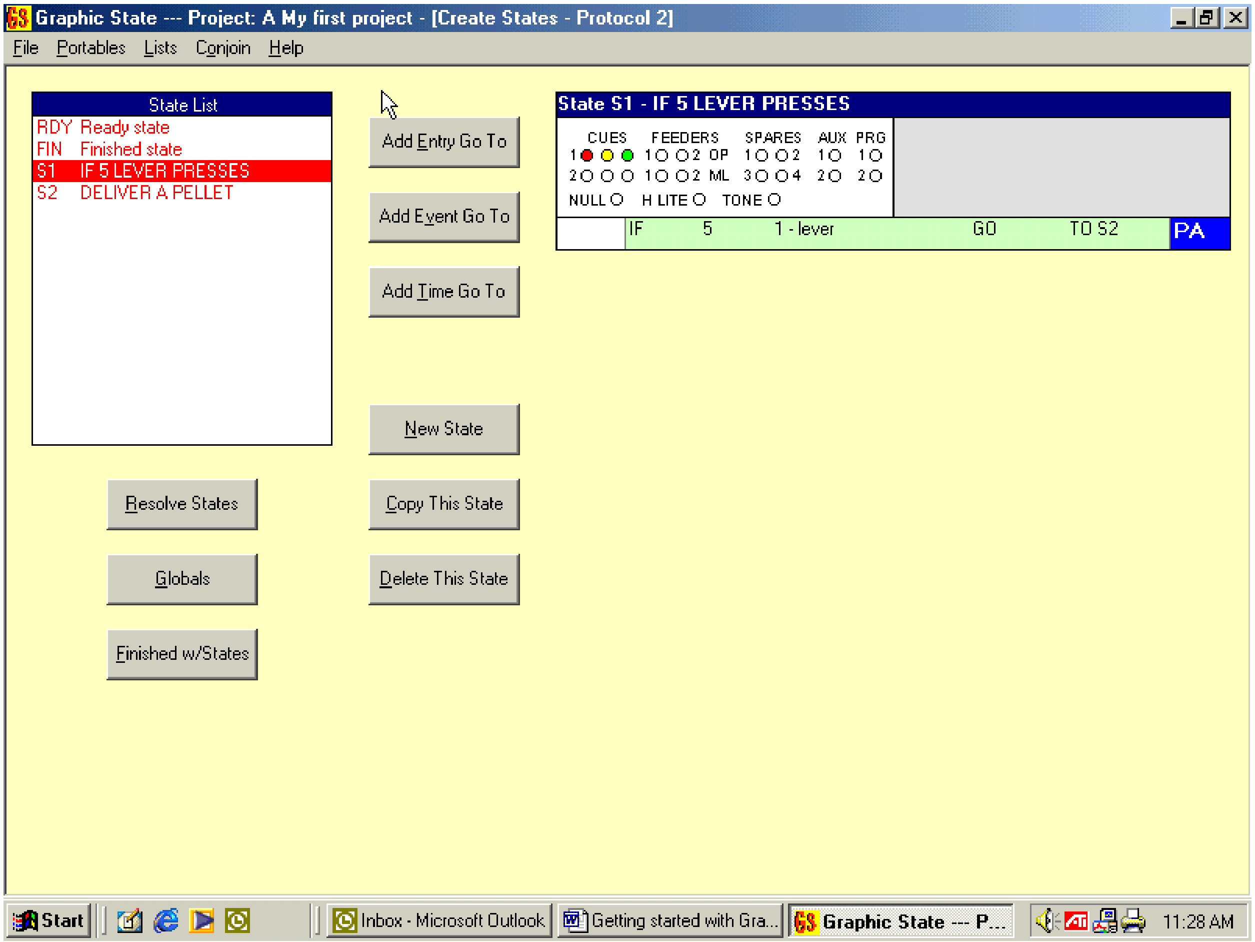


It is as simple as that to change a CRF schedule to an FR5 schedule. However, we may want to make some other adjustments to the protocol as well. In particular, what should happen if the lever is pressed during the 5-second periods while the magazine light is on?

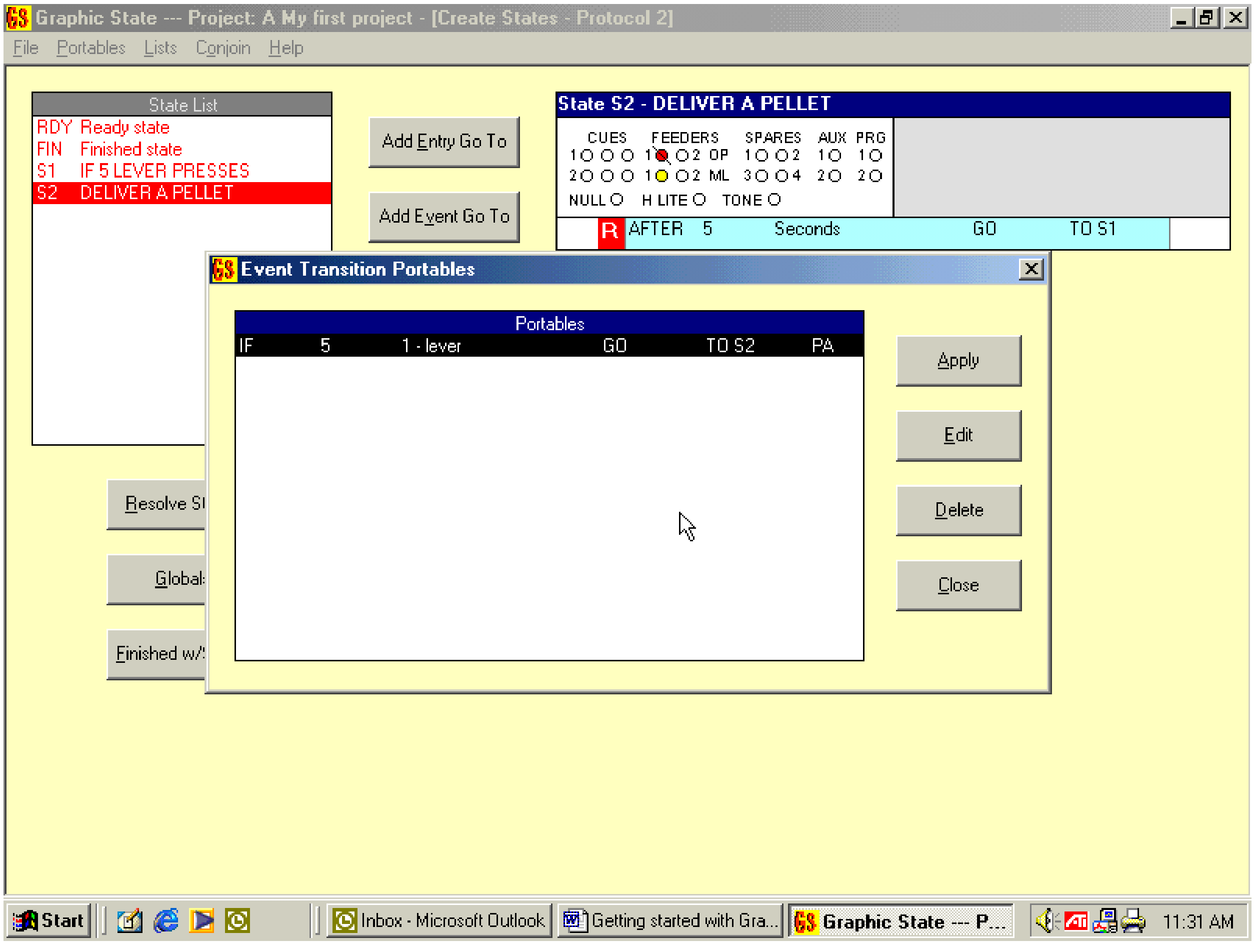
As we have the protocol coded, these responses will be ignored (although they *will* be recorded and we can see that they were made when we do data analysis). Perhaps this is how we want our experiment to run. Lever presses will be counted towards the FR5 requirement only when the triple cue lights are on and the magazine light is off. However, let's assume that we want ALL lever presses to count towards the ratio requirement.

To make this logic change, we want to add a statement to S2 to test lever press responses against the ratio requirement. In addition, we want to specify that this is the *same* FR5 that we are testing in S1. We will do this by making the test for 5 responses a "portable" and then using it in both states.

Select S1 from the State List and then double click on the green IF statement to make it larger and available for changes. At the far left of the statement, there is a check mark next to a large red R. This specifies that the accumulated count of lever presses to be checked against the required 5 should be reset to 0 every time the protocol enters this state. We no longer want to do that, so click on the check box to unselect this option. There is another check box just before the OK button that is labeled, in blue, PA. This stands for "Portable A" (the first available portable designation). Click in that check box and then click OK.

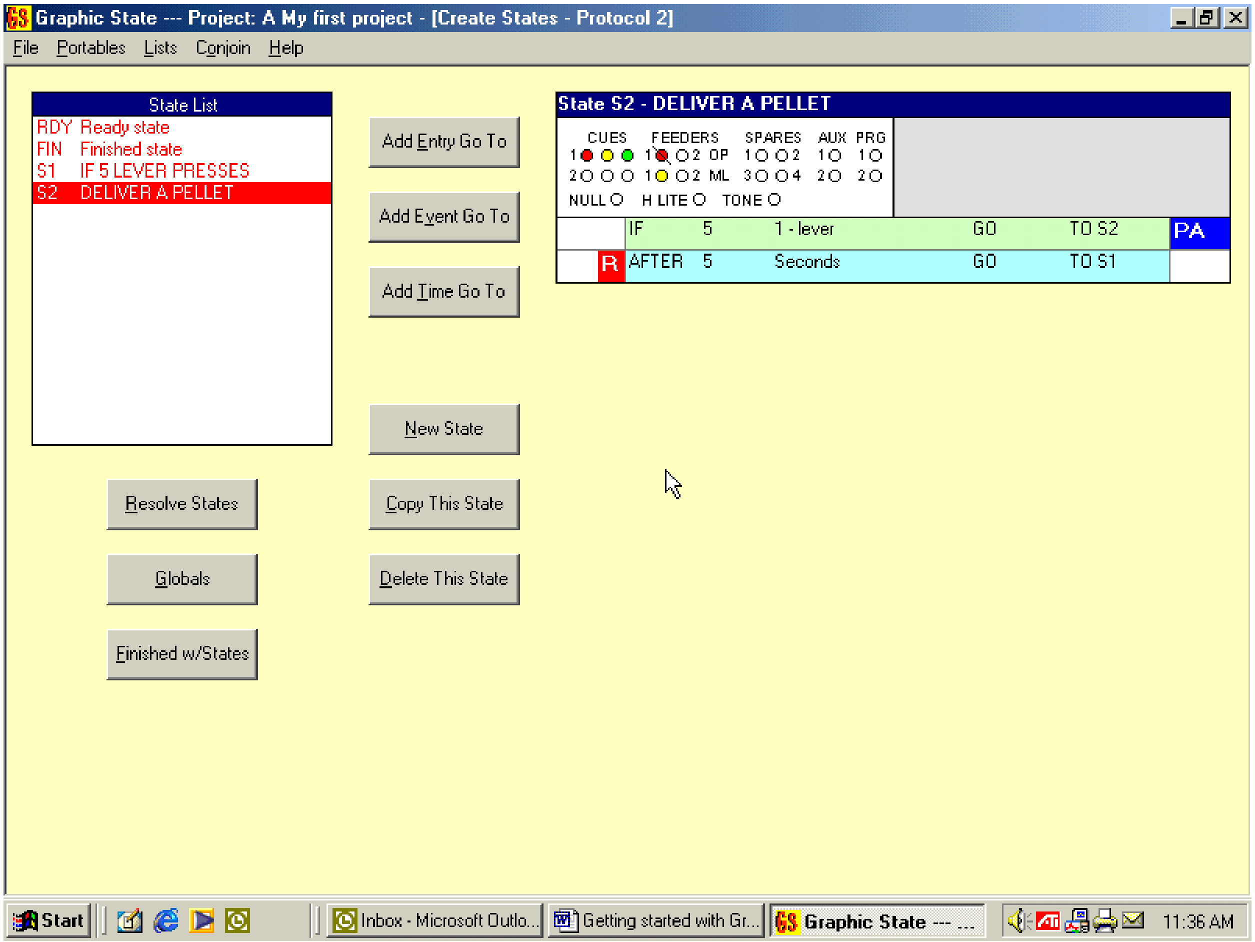


Now select S2 from the State List. Use the Portables pull-down menu at the top of the screen and select the option "Event Portables." This will show you a list of all of the event-type portables that have been defined; at the present time there is only one.



Click on the Apply button to include a test of this portable event in S2. Click on OK to include it without any changes – although in other circumstances with a portable, you may want to change the GO TO location.

Since responses in S2 now also count towards fulfillment of the FR5 requirement, let's turn the triple cue lights on in this state as well. See if you can remember how to do this.

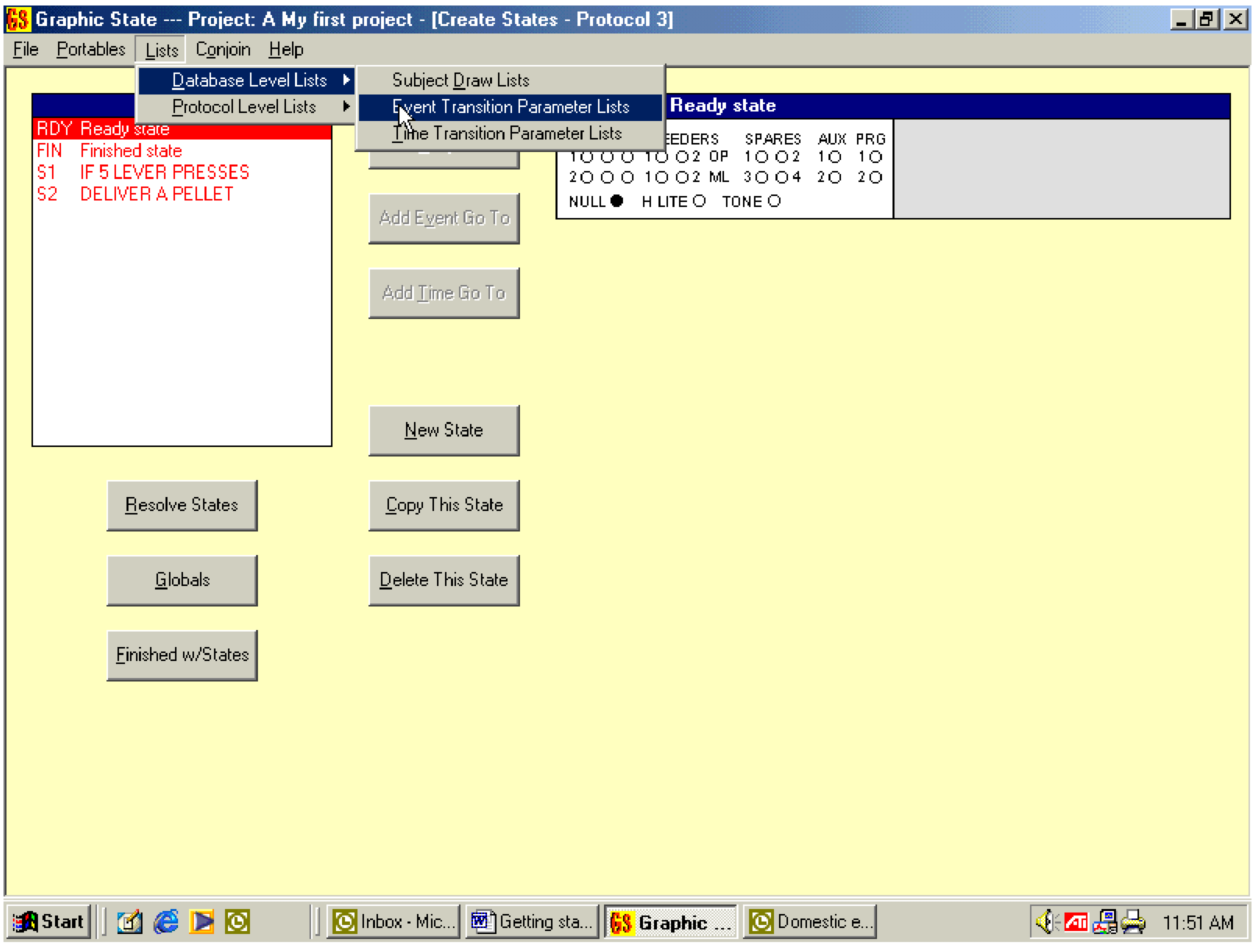


Finally, click on Resolve States to verify that the protocol syntax is correct. We now have a protocol that will deliver a pellet and start a 5-second period with the magazine light illuminated after every 5 lever presses – even if they occur during a prior reinforcement period. Click Finished w/States and then Finished to return to the main screen.

Lesson 3 – Using a list to create a VR schedule

Now let's copy our FR5 protocol to create a VR5. Start the program and select our sample Project (unless it is still active because you are continuing from the last lesson without leaving the program). Select Edit/Review Experiment Protocol and then highlight Protocol 2 and click on the Copy button. Change the name of this protocol to "VR5" and click on State Creation.

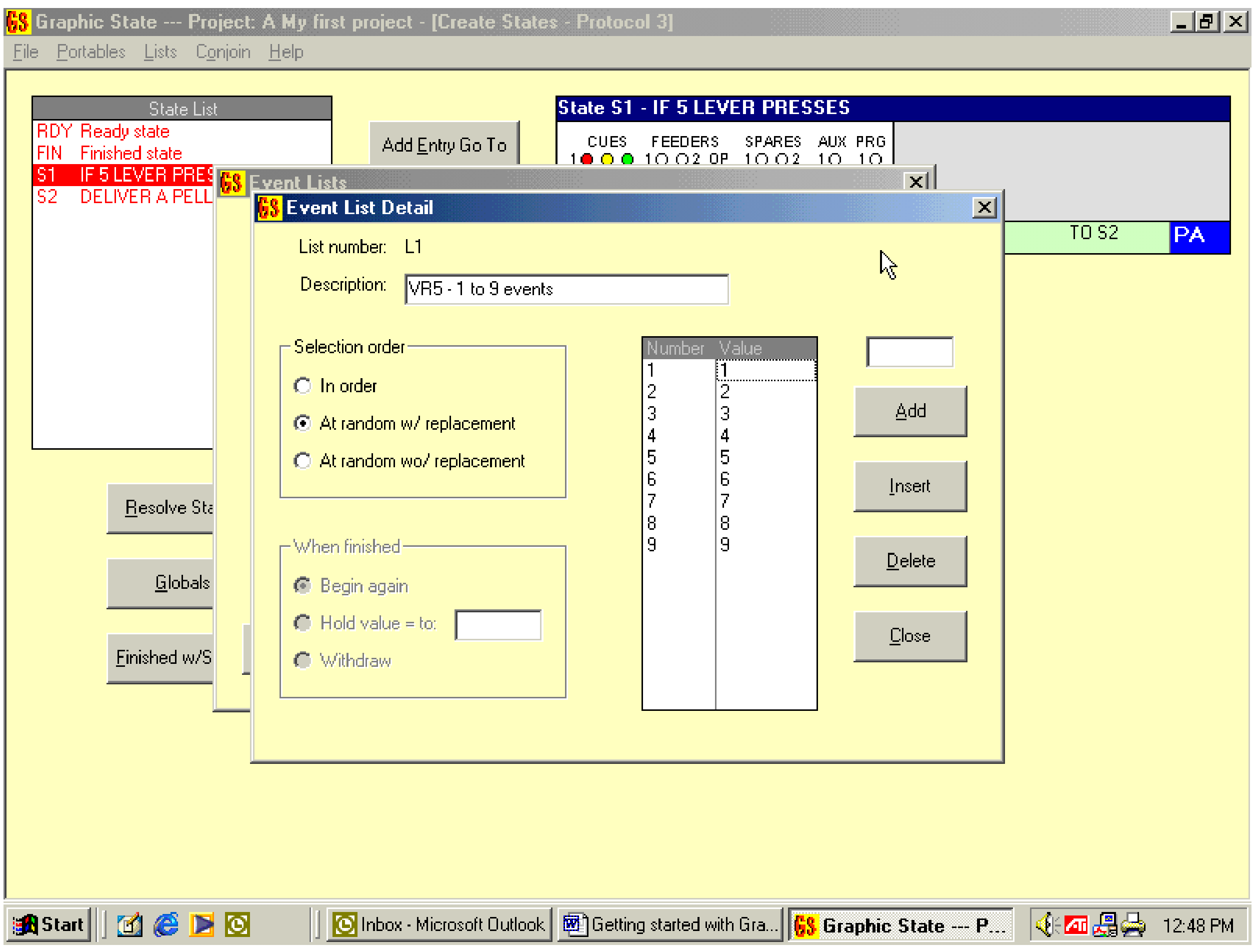
We are now going to change the references to exactly 5 lever presses with references to a list of possible ratio requirements. From the Lists menu at the top of the screen, select Database Level Lists and then Event Transition Parameter Lists. We are going to create a list of ratio requirements that can be used not only in this protocol, but also in any others that might need it in this Project/Database.



This will bring up a screen where you should click on Add to create a new list. Enter a list number ("1") and a name ("VR5 – 1 to 9 events") and then click on Add Events. We will be implementing a VR5 with a list of 9 possible values (from 1 to 9) each with an equal probability of being chosen, at random, with replacement after each prior trial has been completed.

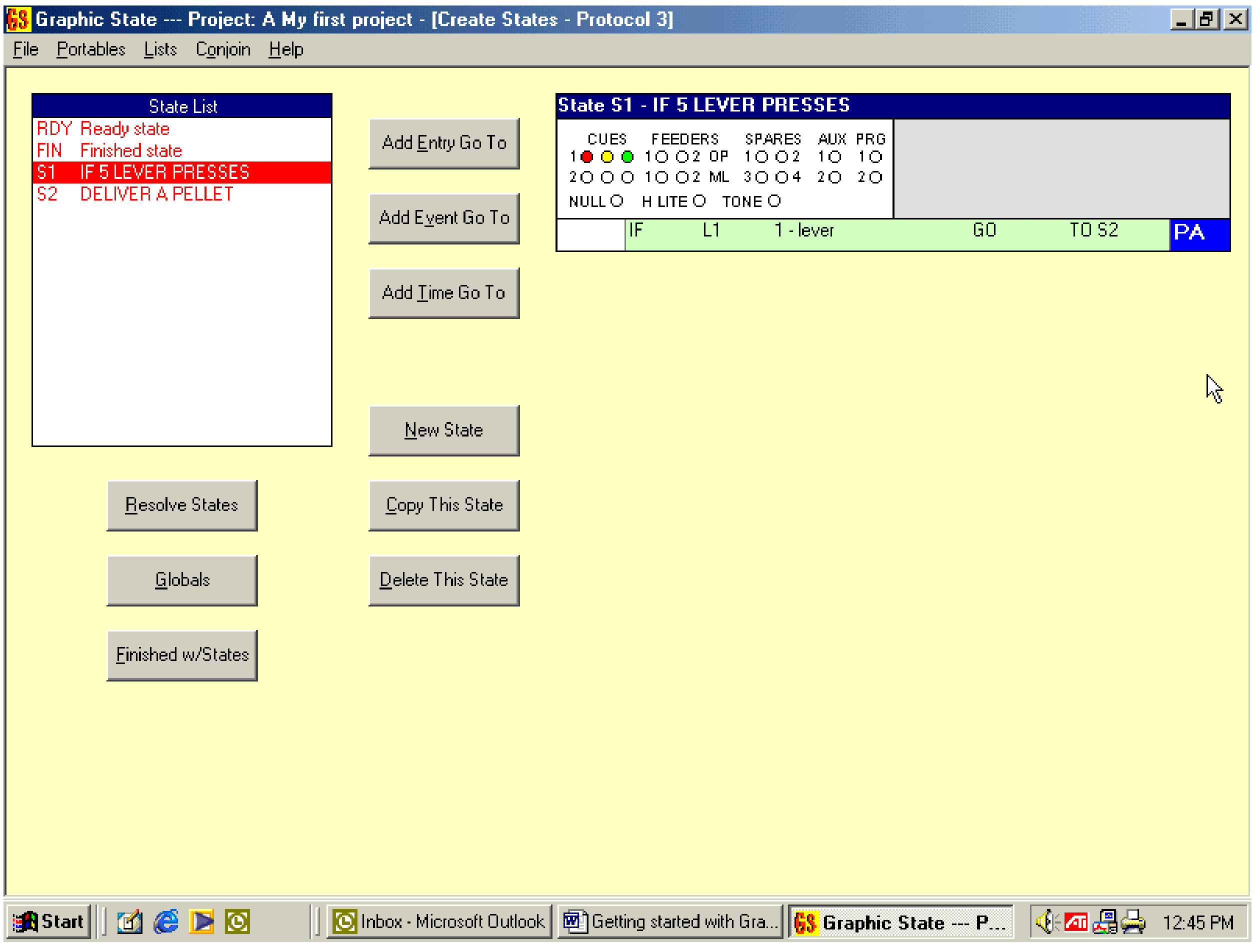
First, enter a "1" where the cursor is blinking and then click on the Add button. Repeat this process for the numbers 2 through 9. Next select the "At random w/

replacement" option under "Selection order". Notice that the other choices here are to select from the list at random without replacement (which will use all of the choices once before used choices are available again) and to use the list in order. In either of these last two options, the list can be exhausted. In that case, you can specify under "When finished" what should happen. The choices are to "Begin again" (to reuse the list), use a constant value after that, or to completely withdraw the statement that tests the value. In the case we have selected (at random with replacement), the list is never exhausted and we don't need to specify anything else.



Click on the Close button to save the list.

To use this list instead of the fixed value "5", we need to change the Event Portable in which the ratio requirement is tested. (If the test were not part of a Portable, we would change it directly in the state where it was used.) From the Portables menu at the top of the screen, select Event Portables and then click on the Edit button. Change the constant “5” after the IF to read “L1” and then click OK. Click on the Close button. Your screen should now look like this:



If you select S2 from the State List, you will see that the constant “5” has been replaced by the reference to L1 in that state as well. Now, each time a value from the list is “used up” (by the appropriate number of responses being recorded), another value from the list will be selected at random. This will continue until the run ends after 30 minutes.

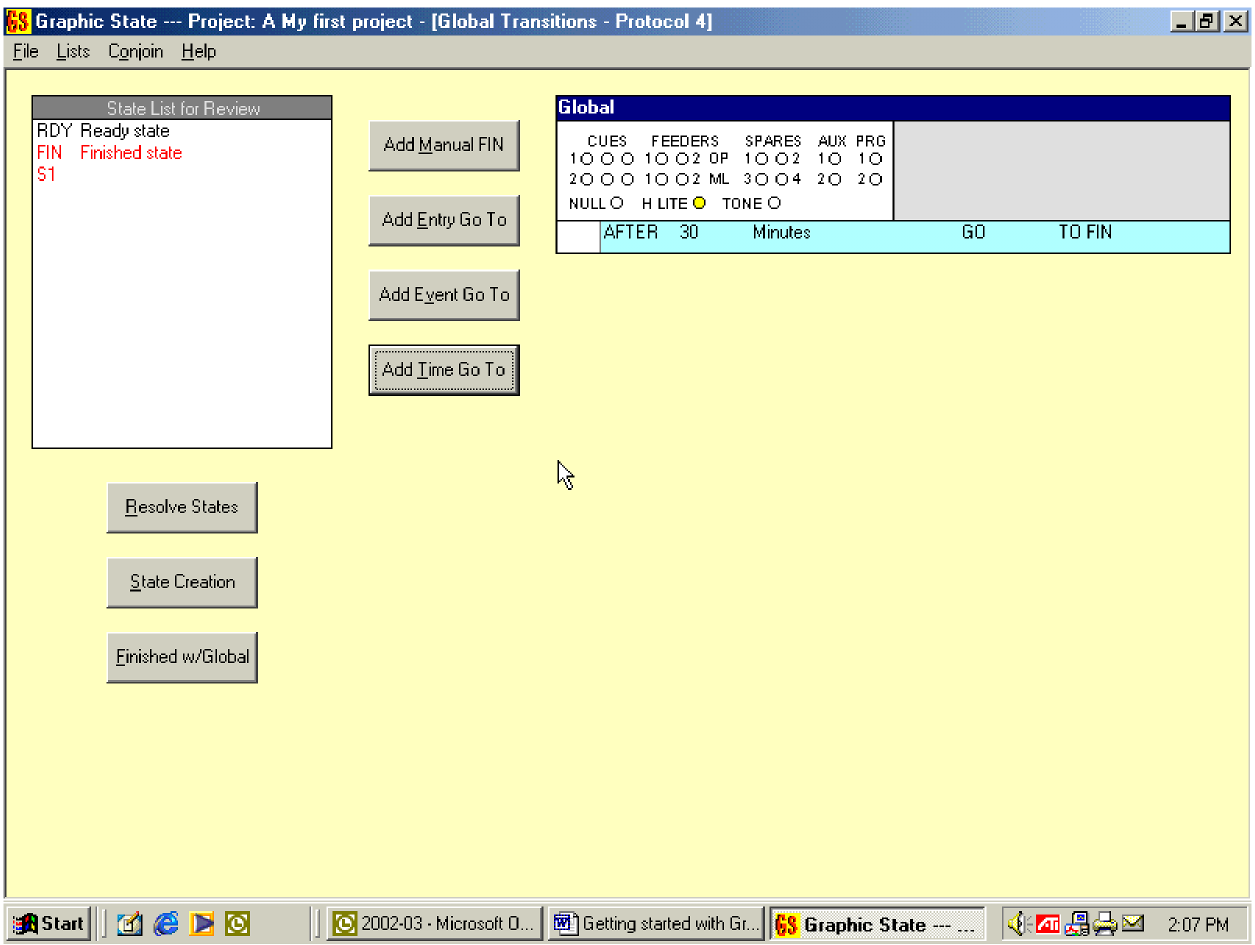
Don’t forget to Resolve States before leaving this protocol (Finished w/States and then Finished).

Lesson 4 – Creating an FI or a VI schedule

Another common schedule type is the interval schedule (either a Fixed Interval or a Variable Interval). Let’s start with an FI. From the File menu, select Create

Experiment Protocol, accept the default Protocol number (4) and enter the Description "FI5".

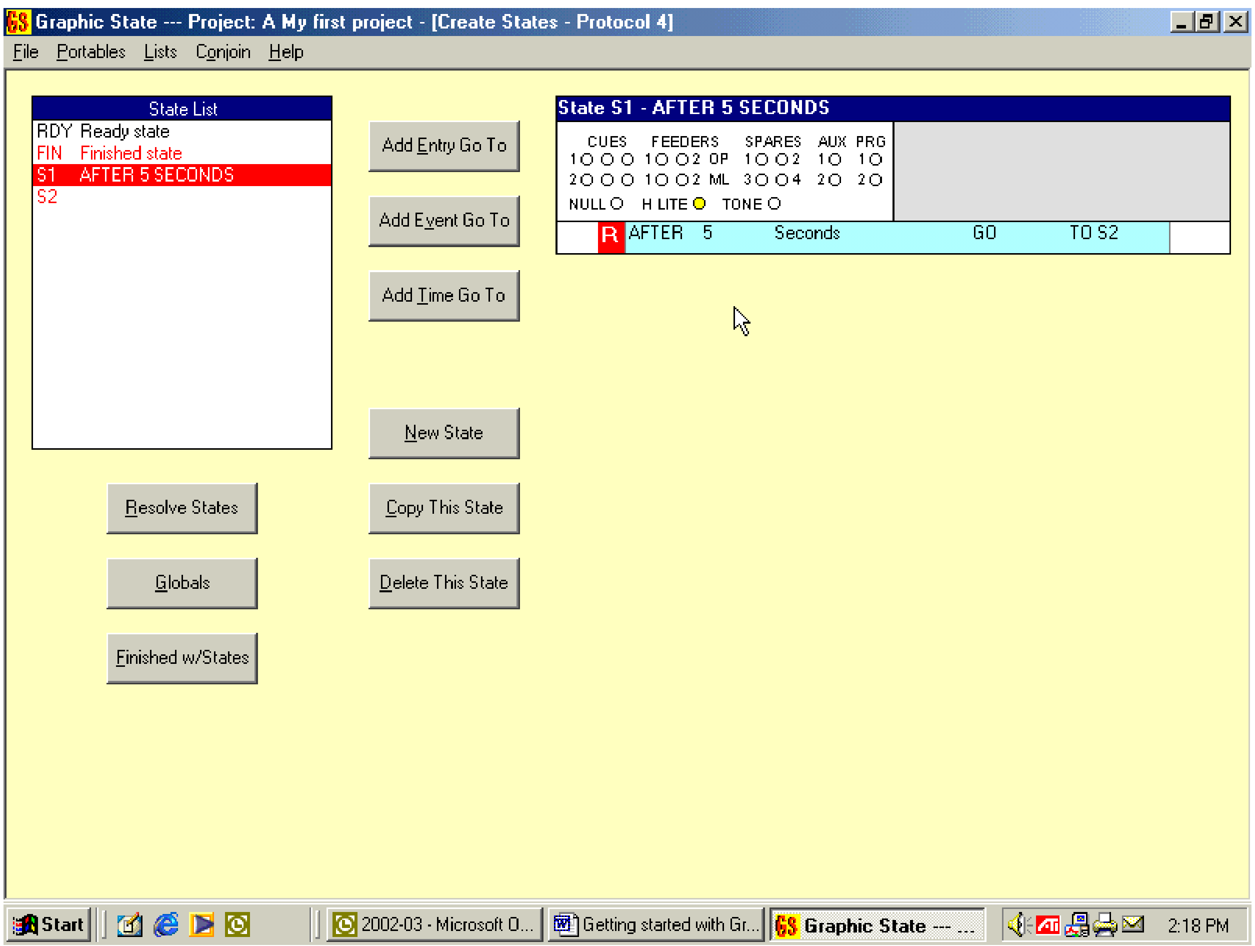
For this example, let's start by clicking on the Globals button to enter a run termination time and also to enter a global stimulus. Double click on the stimulus graphic, click on the "H LITE" button and then click OK. This will turn the house light on in *all* states except where we explicitly choose to turn it off. Now use the Add Time Go To button to add the 30-minute termination criterion.



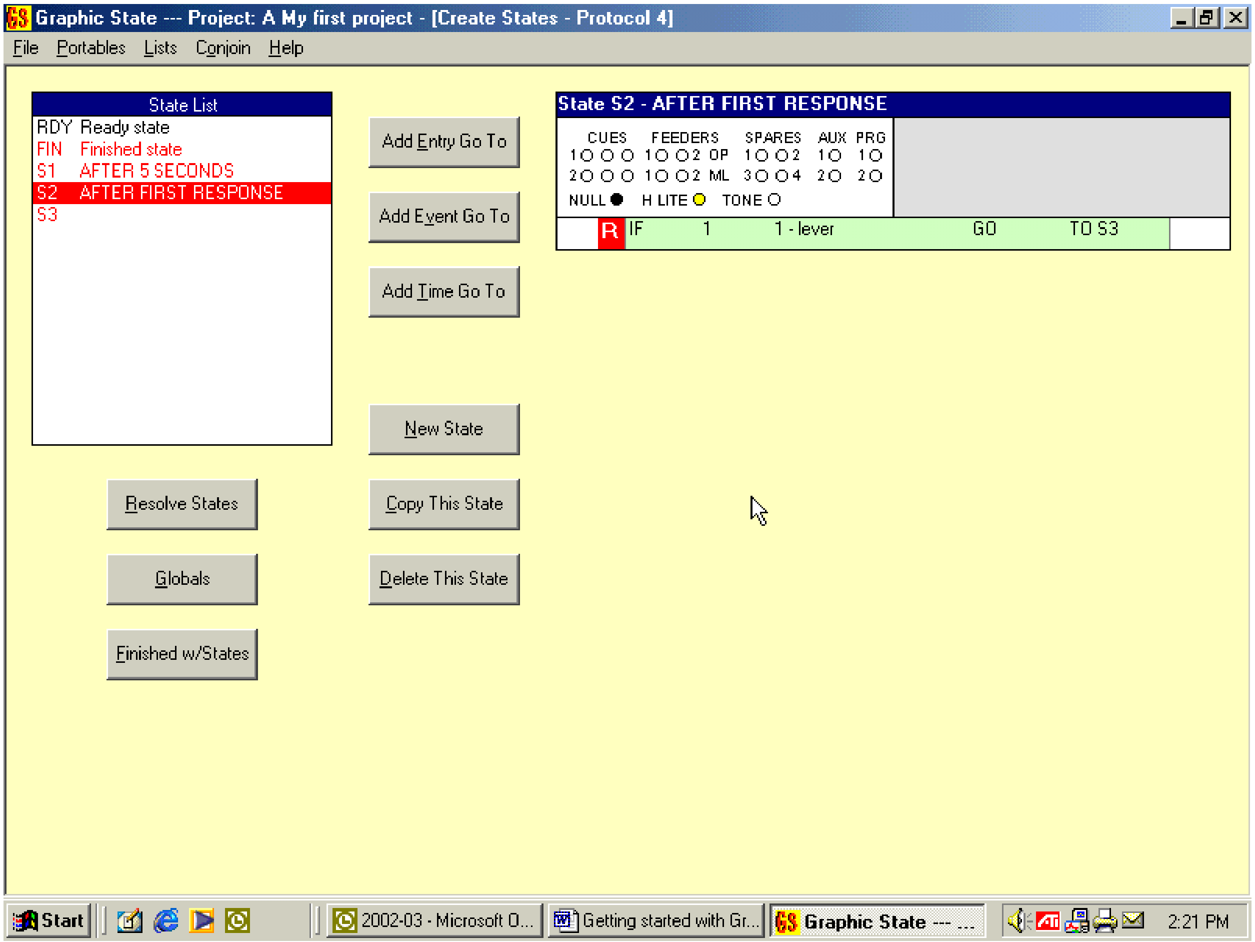
Click on the State Creation button to move to the Create States screen.

The nature of an interval schedule is that the first response *after* a time interval has passed will be followed by the delivery of a reinforcer. We will create a protocol that stays in S1 until the interval has passed, checks for the first response in S2 and delivers the reinforcer in S3. Because we do not want the magazine light to be an additional time cue, let's not use it at all in this protocol.

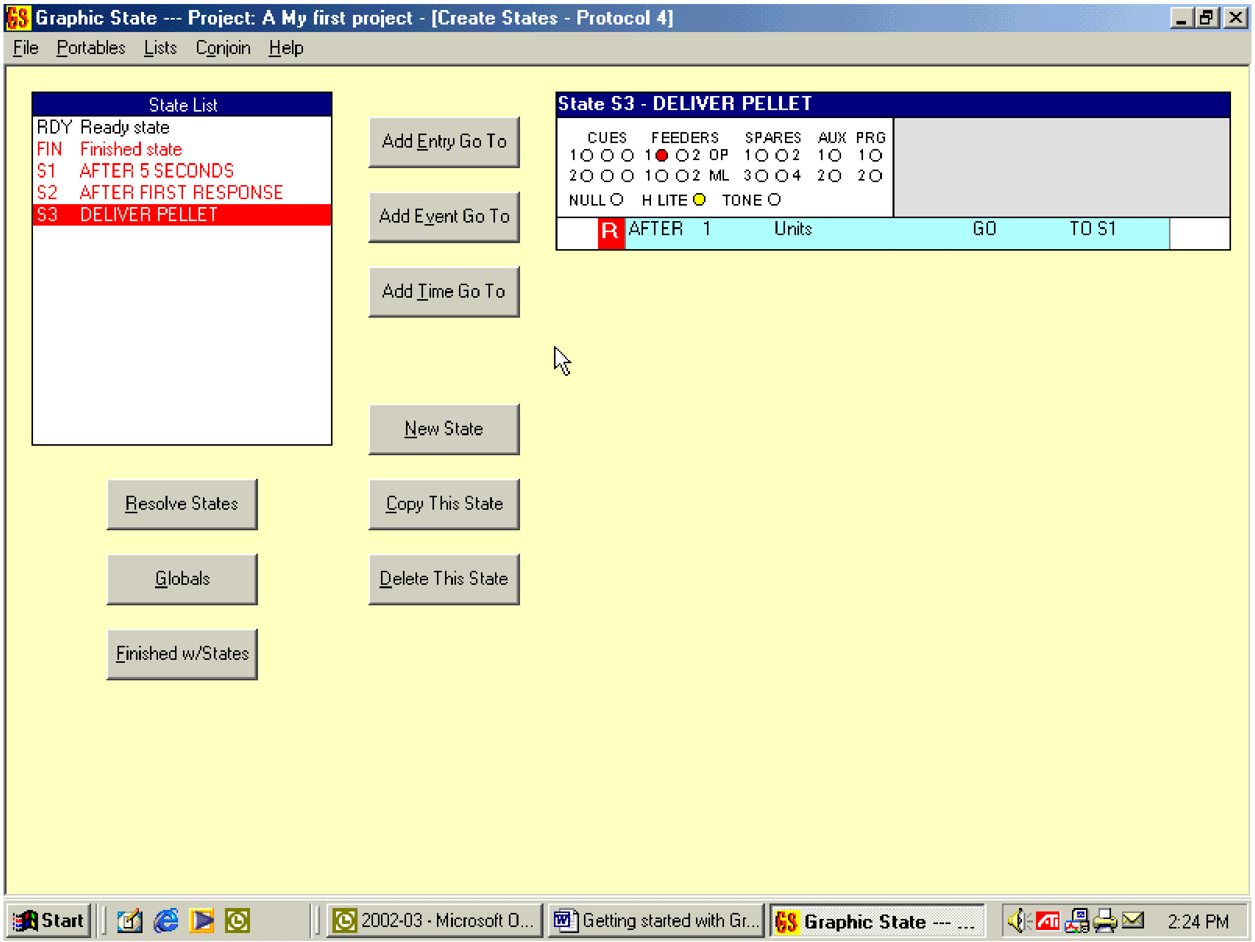
Select S1 from the State List. Notice that the house light button is already yellow because we turned it on in the Globals screen. We don't need any other stimuli in this state, so there is no need to double click on the stimulus graphic. We do want to give S1 a name, however. Select the blue area at the top of the stimulus graphic and type in a name/description of "AFTER 5 SECONDS". Click OK. Then click on the Add Time Go To button and specify AFTER 5 seconds GO TO S2.



Select S2 from the State List. Add a state name of "AFTER FIRST RESPONSE" and use the Add Event Go To button to specify IF 1 lever response GO TO S3.



Select S3. Add a state name of “DELIVER PELLET”. Turn on the OP button for Feeder 1 and then add a time parameter that specifies AFTER 1 unit GO TO S1.



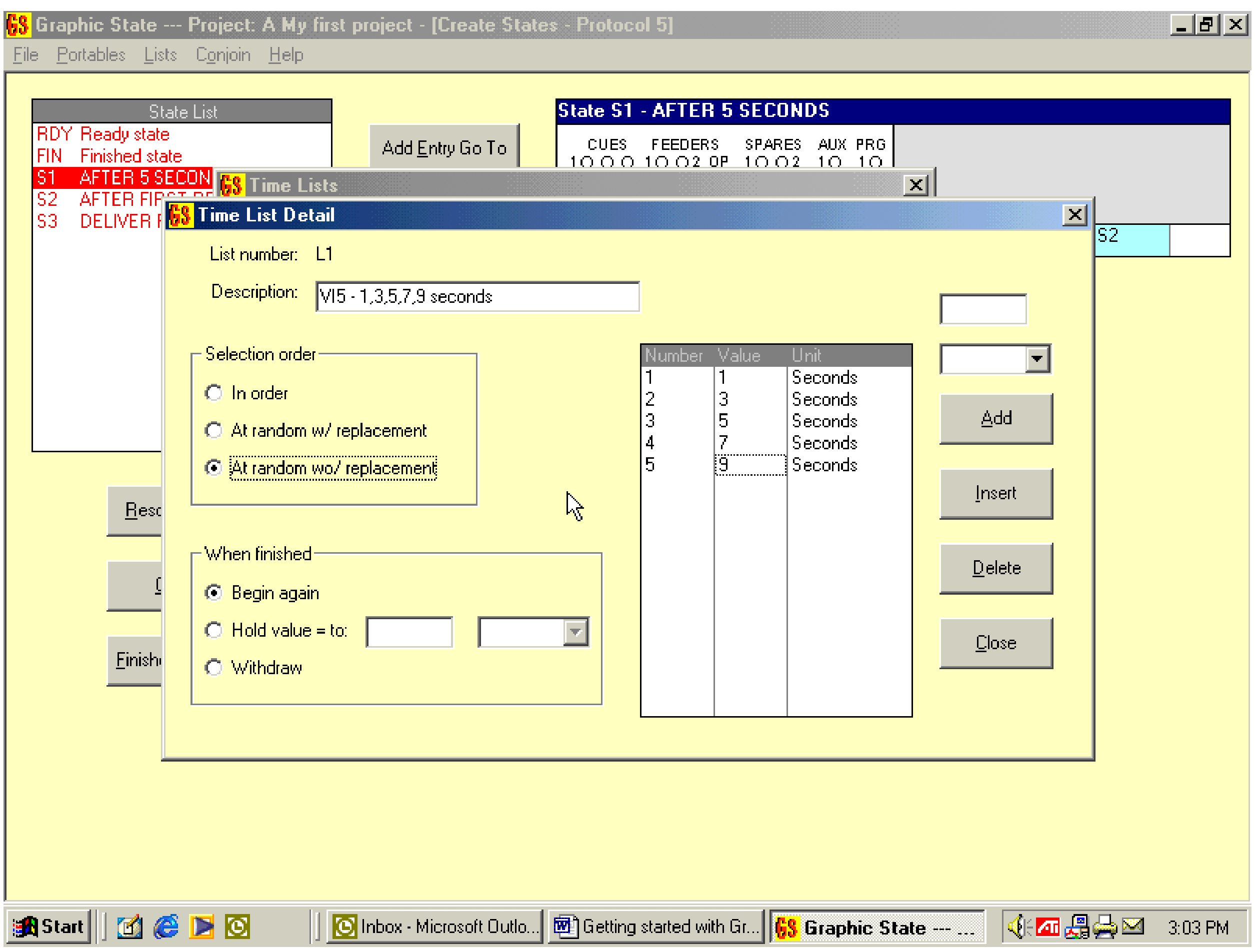
We do not need to use the stimulus pulsing feature because we will only remain in S3 for a single sampling unit anyway.

Click Resolve States and then exit back to the screen with the Graphic State logo.

Now let's create a VI schedule. From the File menu, select Edit/Review Experiment Protocol. Highlight Protocol 4 and click on the Copy button. Change the description to read VI5 and then click on the State Creation button.

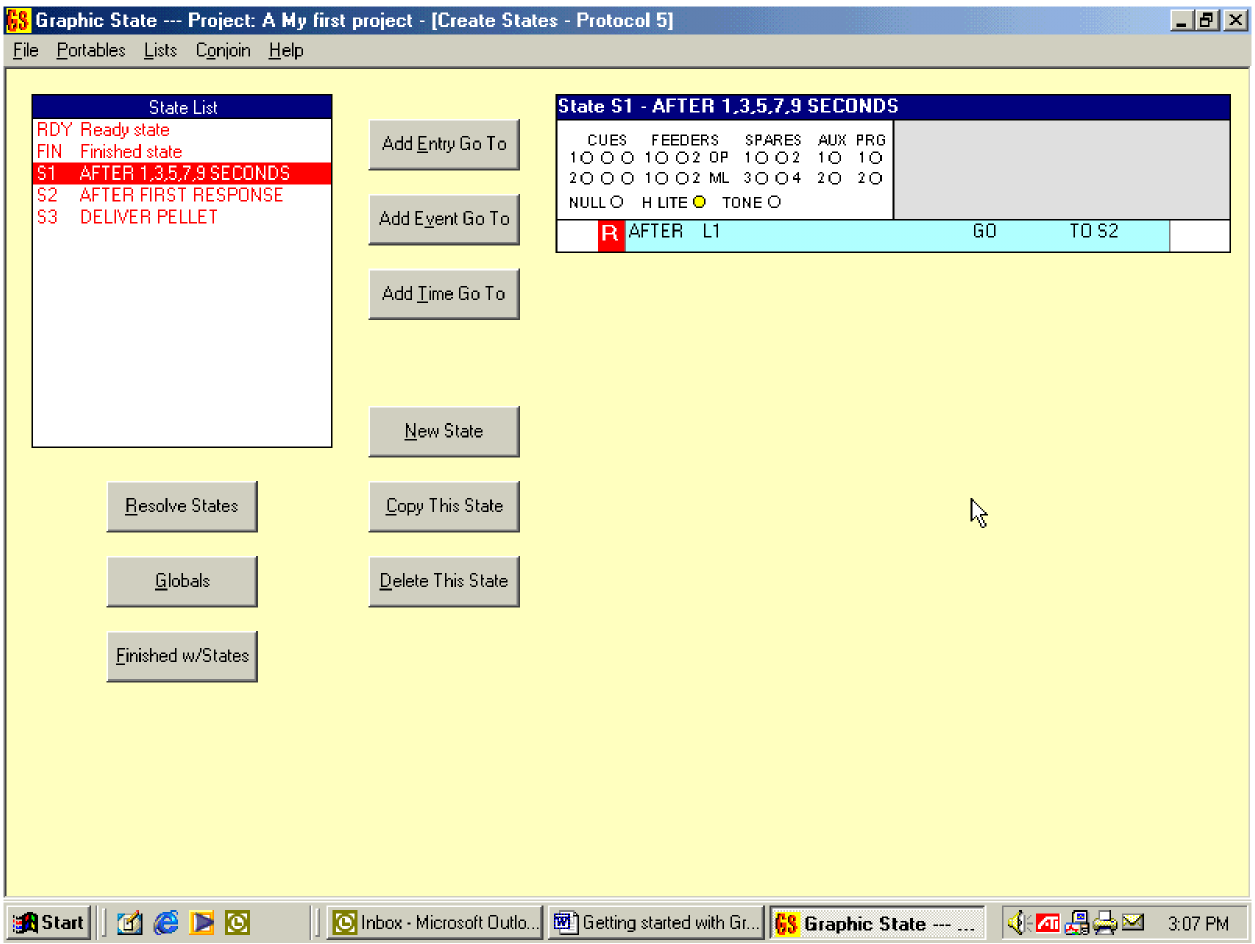
Select S1 in the State List. We want to change the reference to a constant 5 seconds to a reference to a list of times with a mean of 5 seconds. This is just like the ratio requirement list we made except that it will refer to time intervals. Before we can use the list in S1, we must create it. From the Lists menu at the top of the screen, select Database Level Lists and then Time Transition Parameter Lists. Use the Add button to create a new list. This list may be called L1 without any confusion with the Event List we made earlier because we can't use the wrong type at any point in a protocol. Give the list a name such as "VI5 – 1,3,5,7,9 seconds" and click on Add Times.

We are going to make a list with five time intervals (1 second, 3 seconds, 5 seconds, 7 seconds and 9 seconds) in it. There are many ways to implement a VI5 seconds (as long as the *mean* time is 5 seconds), and this example illustrates one of them. Enter a "1" where the cursor is blinking and then select "Seconds" from the pull-down list just below it. Your choices there are Units (sampling units – 100 ms in our case), Seconds and Minutes. Click the Add button to put "1 second" in the list. Repeat this process to add the other time intervals to the list. After you have entered all five values, select "At random wo/ replacement" under "Selection order" and "Begin again" under "When finished".

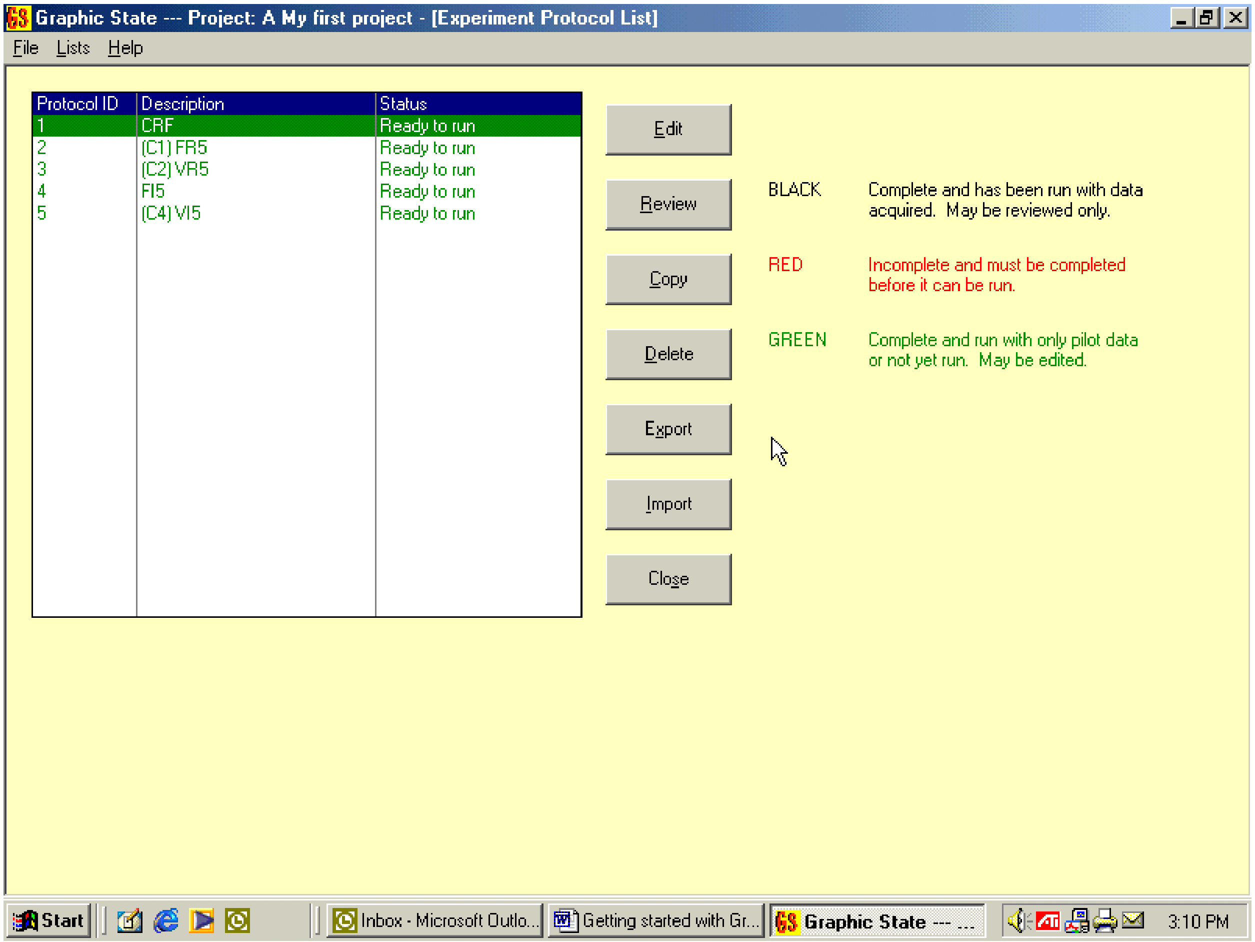


We have now specified that the time interval for each trial will be selected at random from this list, but that all five values will be "used up" before any particular time is used again. Click on the Close button (and then another Close button) to save the list.

Double click on the time specification line under the stimulus graphic in S1. Change the constant "5" to the list name "L1" and click on OK. Don't forget to change the name field for S1 as well.



Resolve the states and exit from the protocol. Go to the Edit/Review Experiment Protocol screen and see that you now have five protocols that all have a status "Ready to run".

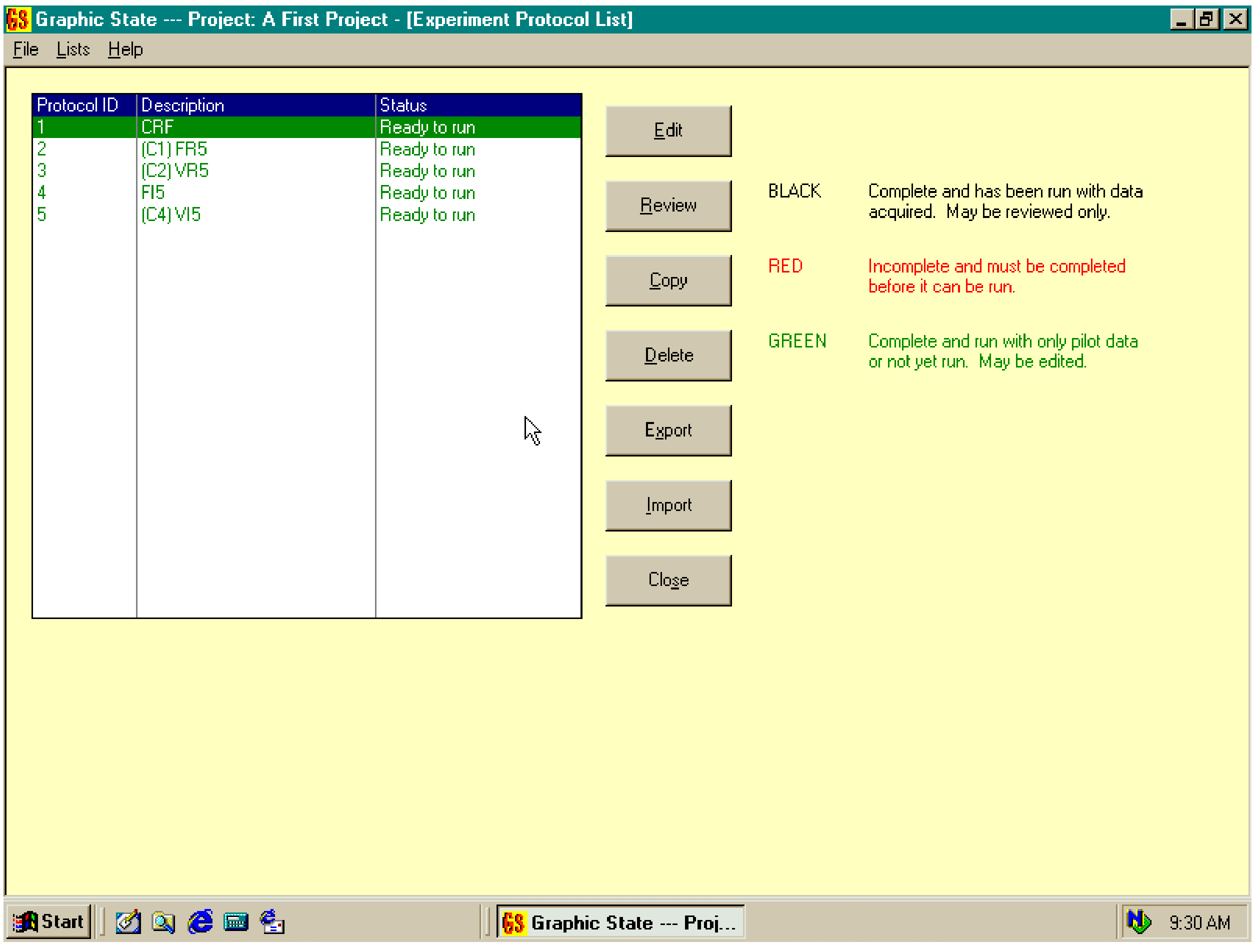


We will discuss running sessions and analyzing data in later portions of this Tutorial. Creating the specifications for analyzing data (Analysis Structures and Analysis Elements) is, or can be, part of the initial creation of the protocols. However, this can also be done later (even long after data have been collected) – and it may be less confusing to discuss it after some data are available for illustration.

Running an experimental session

Now we can run an experimental session and gather some data. You will either need a cooperative rat or mouse for this, or you can take the place of the animal for now. All you need to run the examples we have created is one lever (plugged into "SWITCH 1" on the ECB) and a pellet feeder and trough (plugged into "FEEDER 1"). You don't even need the feeder or trough if you are taking the place of the animal; the feeder light on the front of the Linc will be your reinforcer. If you don't have a lever, you can use the test Probe to touch the silver "TEST" area next to "SWITCH 1" instead of pressing the lever.

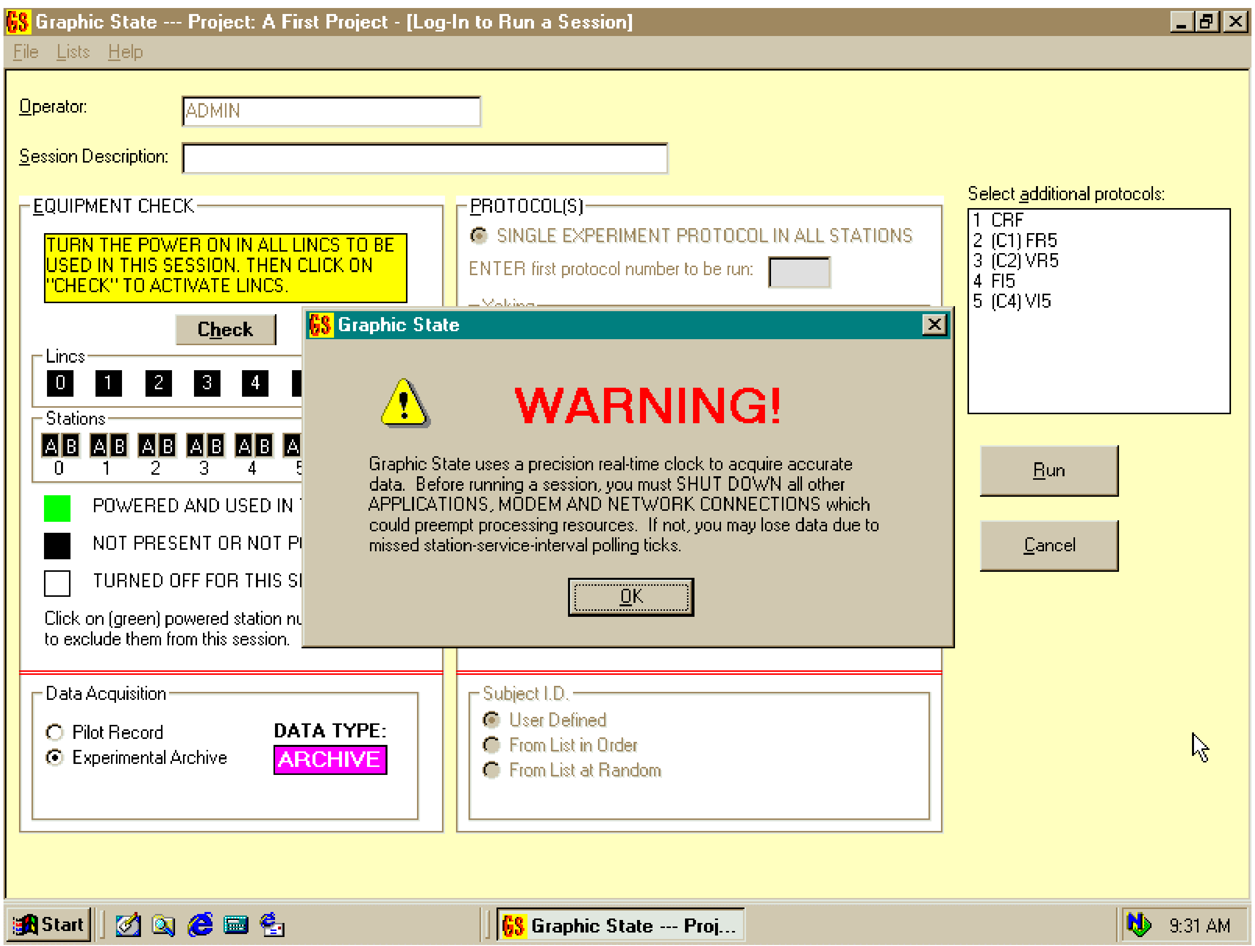
Start up the Graphic State Notation program and select the A_First Project. Select "Edit/Review Experiment Protocol" from the File pull-down menu just to look once more at what protocols are available.



Notice that all five are listed in green with the Status "Ready to run". This means that they have been checked by the "Resolve States" procedure and they appear to be complete, but they have not yet been run with "real" data. Either they have not been run at all (which is the case right now), or they have been run only with "pilot" data (which we will discuss shortly). Once a protocol has been run with real data, it cannot be modified, except to add data analysis specifications, because logic modification would invalidate the data that have already been collected. A protocol can always be copied and the copy can be modified if necessary.

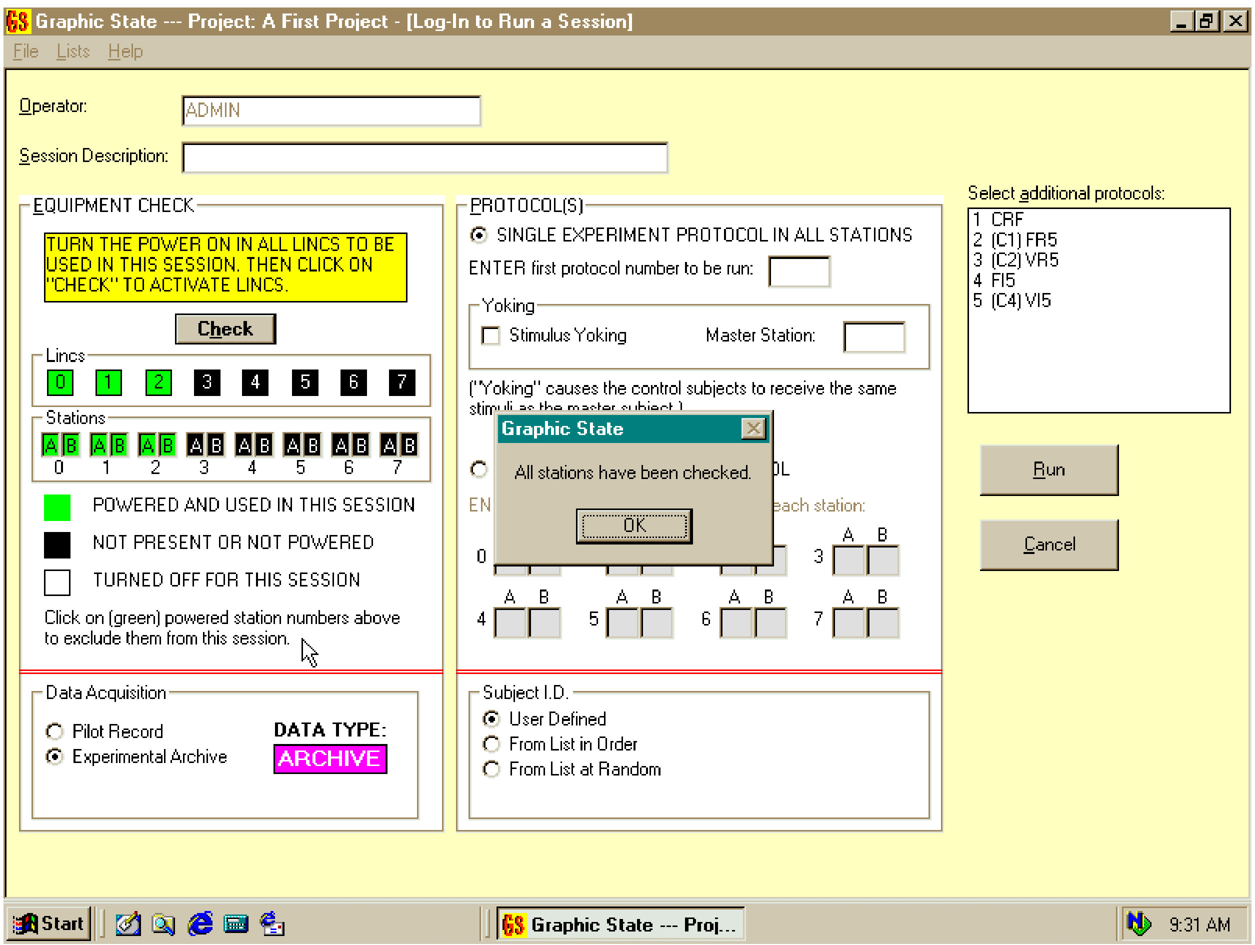
Before I continued with this example, I edited all of the protocols to change the Global times from 30 minutes to 5 minutes. You can do this as well, or leave the times as they are.

From the File menu, select RUN A SESSION. You will always see a warning message that reminds you to close all other programs that might be running and to be sure you are not connected to a network while you run a session.



This looks very frightening, but it is meant to emphasize that Graphic State Notation and the Habitest hardware must have no competition from other Windows programs while protocols are running and data are being collected in order to assure that no communications through the PCI card are missed. This is particularly important if you are running many stations at one time and especially if your sample time is short (20 ms). If the system cannot keep up with the sampling demands, you will get a warning message at the end of the run. Hopefully you will never see this message.

Click OK to move on. Next, click on the Check button to make sure the Lincs are communicating with the software. If something is wrong (e.g., the power is not on to the power base or the PCI card is not installed correctly), you will get a message. If everything is OK, you will see a screen like the one below.



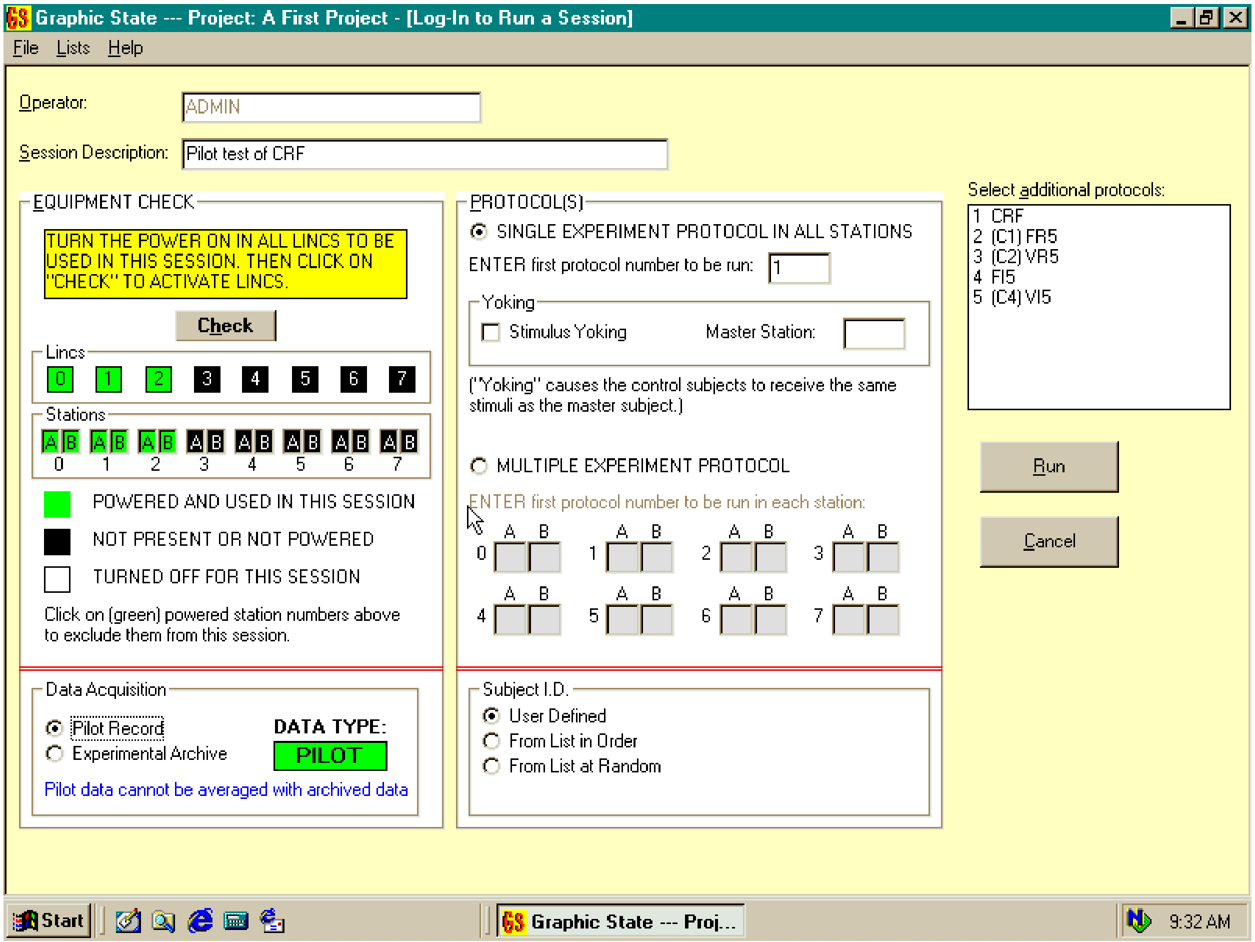
In this particular example, I have three Lincs in my stack and they are all turned on. The three green buttons under “Lincs” tell me this. Since I have my current Project defined as using only one half of a Linc for each station, there are six green buttons under “Stations”. Your screen may show a different number of Lincs and stations, but you should see two stations for each active Linc. Click on the OK button to move on.

Now we will specify some details for this run. Notice that the first field on this screen, Operator, is already filled in with the name "ADMIN". That is because we logged in to the system with the User ID "ADMIN". In order to accurately reflect the actual operator for your real runs, you will want to use the Security features to set up real User ID's and Passwords. You can read the reference manual or just experiment to find out how to do this. For now, leave the Operator field as is.

The next field to fill in is a Session Description. Whatever you enter here, plus the Session number (assigned sequentially by the software) and the Run Date (entered automatically by the software) will be used later to identify the session for data analysis. Enter a description that will help you be sure you know what session it is. For now, enter "Pilot test of CRF" in this field.

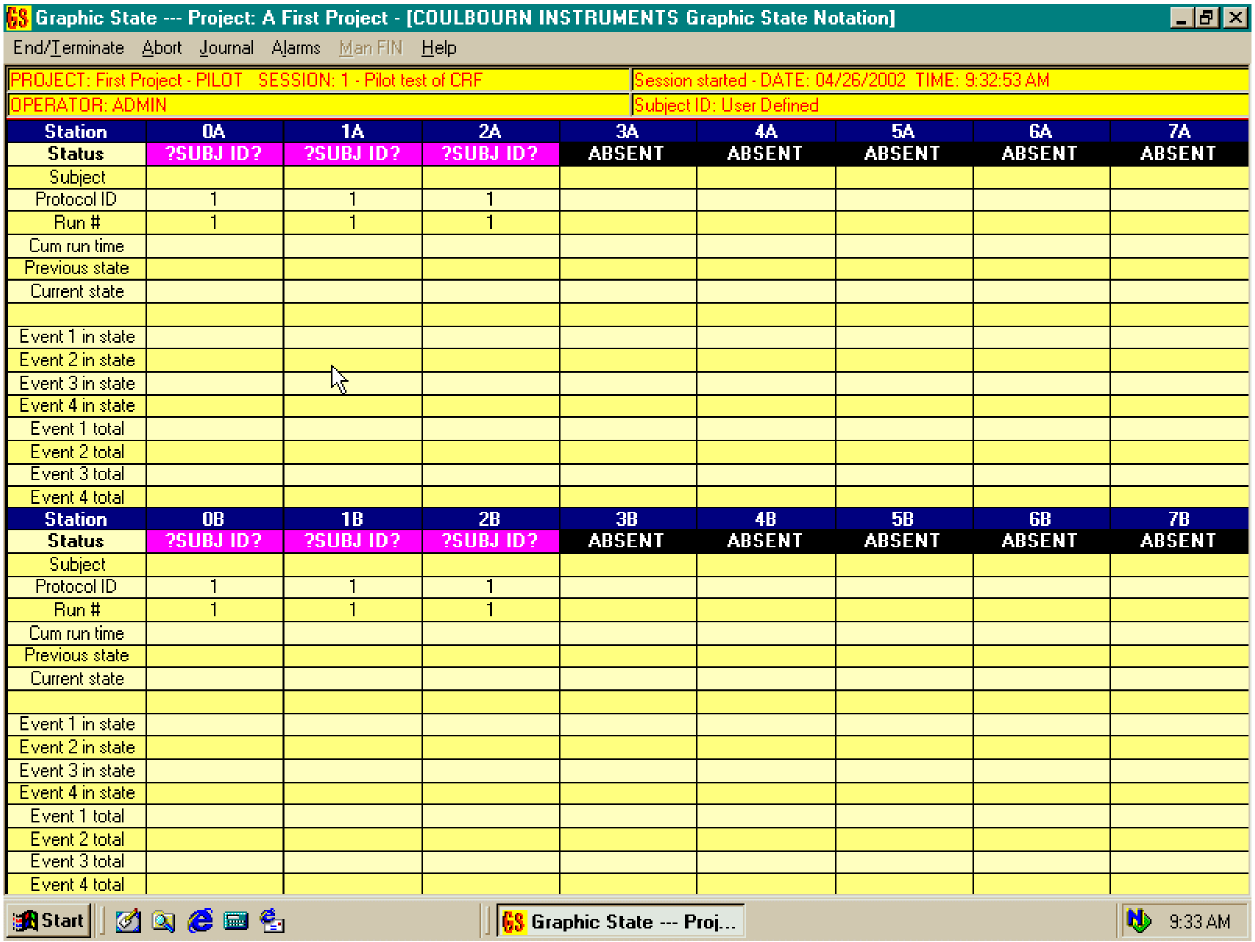
In the middle column of information on this screen, we will specify details about what protocol is going to be run and in what mode. There is a list of available protocols for this project on the right. We will be running the same protocol in all stations and the button to specify this is already selected. Enter the number "1" in the field "ENTER first protocol number to be run".

Let's run this protocol for the first time in "Pilot" mode. In the lower left portion of the screen under "Data Acquisition", click in the check box beside "Pilot Record". The pink ARCHIVE notice will change to a green PILOT notice.



Now click on the Run button.

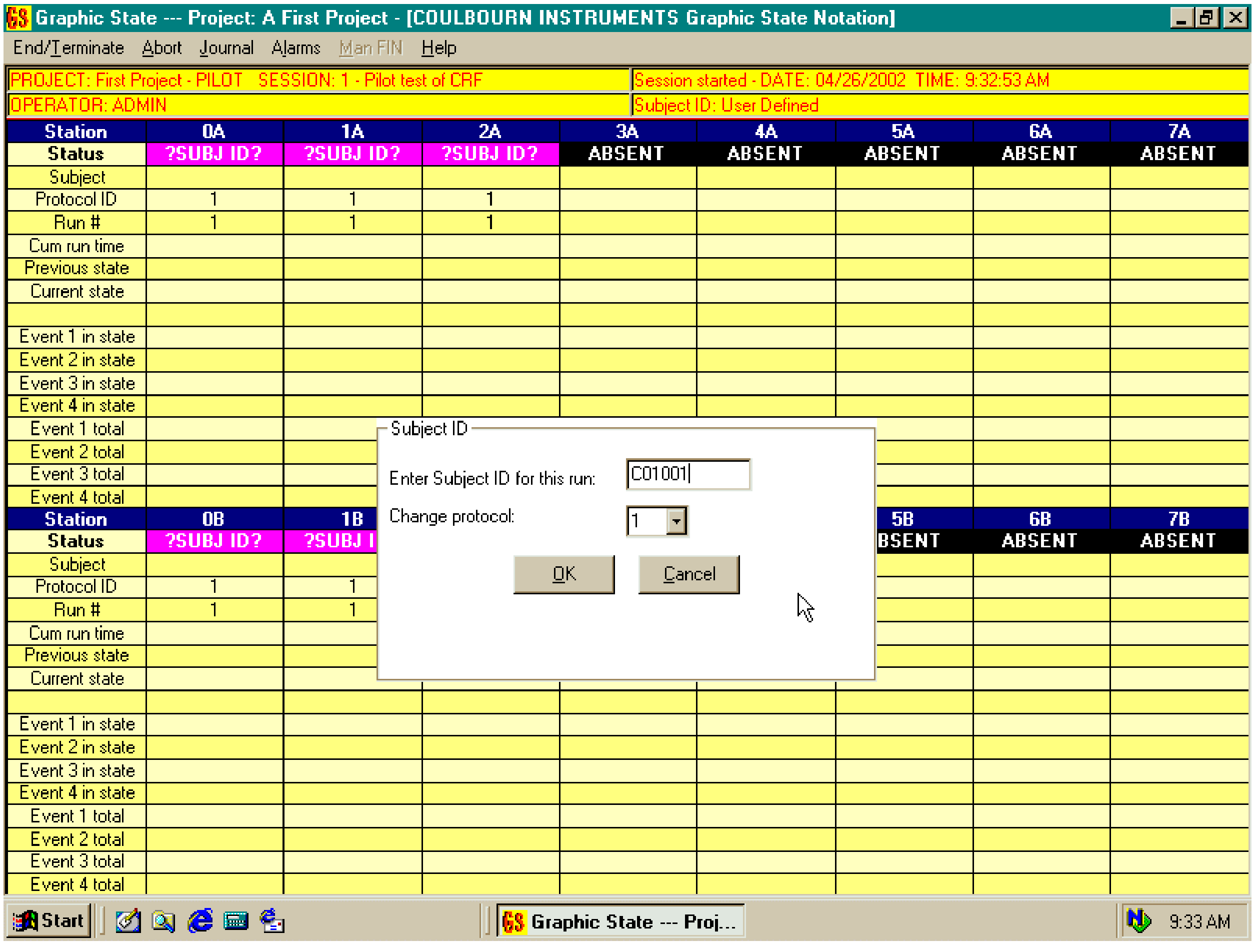
The run screen looks a bit like a spreadsheet. There is a section on it for every possible station given the Linc configuration we have; with 8 Lincs we could have 16 stations. The six that are shown below (because I have 3 Lincs in my stack) have pink headers waiting for an animal ID to be entered. You may have a different number of active stations.



To enter an animal ID, click on the "?SUBJ ID? field for the station you will be using for this run. I will be using Station 0A (the left half of the top Linc on the stack). I will only be running one station for this session, but I could run the same or different protocols in each of the active stations.

The following screen will come up asking for the ID of the animal that is being assigned to this station for this run. There is a way to create a list of animals and to assign animals from the list in order, but for this example we will assign an animal manually. Enter the animal ID "C01001". Using the possible (not mandatory) convention I described earlier, this is animal # 001 belonging to control group 01.

Ignore the "Change protocol" field for now and click on the OK button. If you are using Version 2.102-00 or above, you may use the Enter key instead of the OK button.



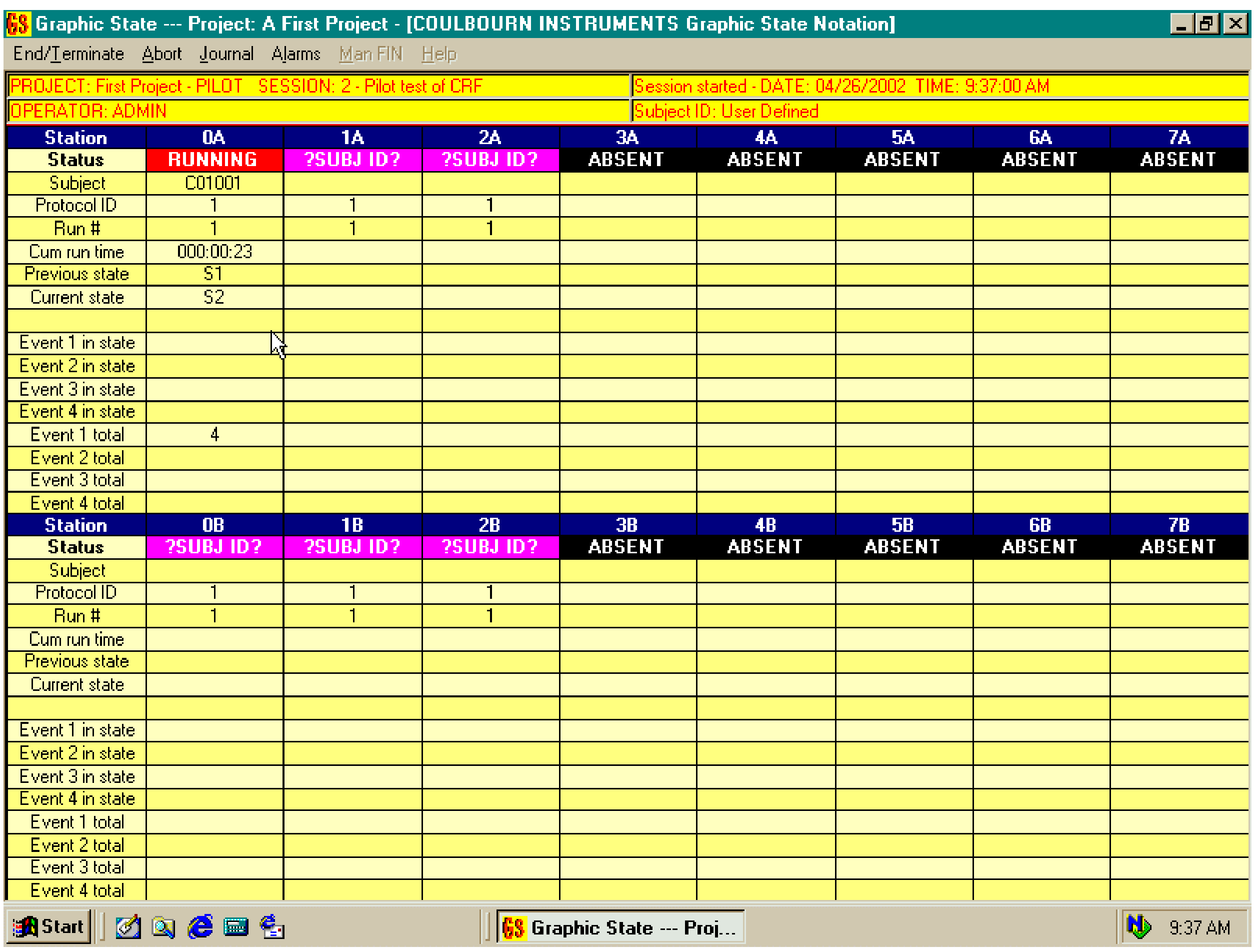
The station header field will change to a green READY and the RDY state in the protocol will be active. We did not specify any stimuli for the RDY state, so this will not be apparent.

If you are using a real animal, place it in the test cage now. If not, put the lever in some convenient place where you can reach it. Click on the READY header to start the run. The header will turn to a red RUNNING status and the run will begin.

If you are using Version 2.102-00 or above, there is a “Start All” option on the menu bar at the top of the screen. Clicking on this will start the run for all stations that are in the READY state.

The protocol will begin in S1 and the first triple-cue light will be operated with all three lights on. If you do not have an actual triple-cue light in the cage, you can at least see the three lights on the front of the Linc. When you, or the animal, press the lever, the feeder will be operated briefly (and the feeder light on the front of the Linc will blink once) and the magazine light (and the corresponding light on the front of the Linc) will come on for 5 seconds while the triple-cue light will go off. The display on the run screen will show that the "Current state" is S2 and the "Previous state" was S1. The count of lever presses ("Event 1 total") will be incremented. This number will be incremented *every* time the lever is pressed, even if the response does not cause a state change. We coded this particular protocol so that lever presses in S2 (while the magazine light is lit) do not cause a state change or the delivery of another reinforcer, but they are recorded nevertheless. If we want to know how many of them *did* result in the delivery of a reinforcer, we can find out during data analysis.

Notice also that the elapsed time since the beginning of the run is shown for each run as "Cumulative run time".



Graphic State --- Project: A First Project - [COULBOURN INSTRUMENTS Graphic State Notation]

End/Terminate Abort Journal Alarms Man FIN Help

PROJECT: First Project - PILOT SESSION: 2 - Pilot test of CRF | Session started - DATE: 04/26/2002 TIME: 9:37:00 AM

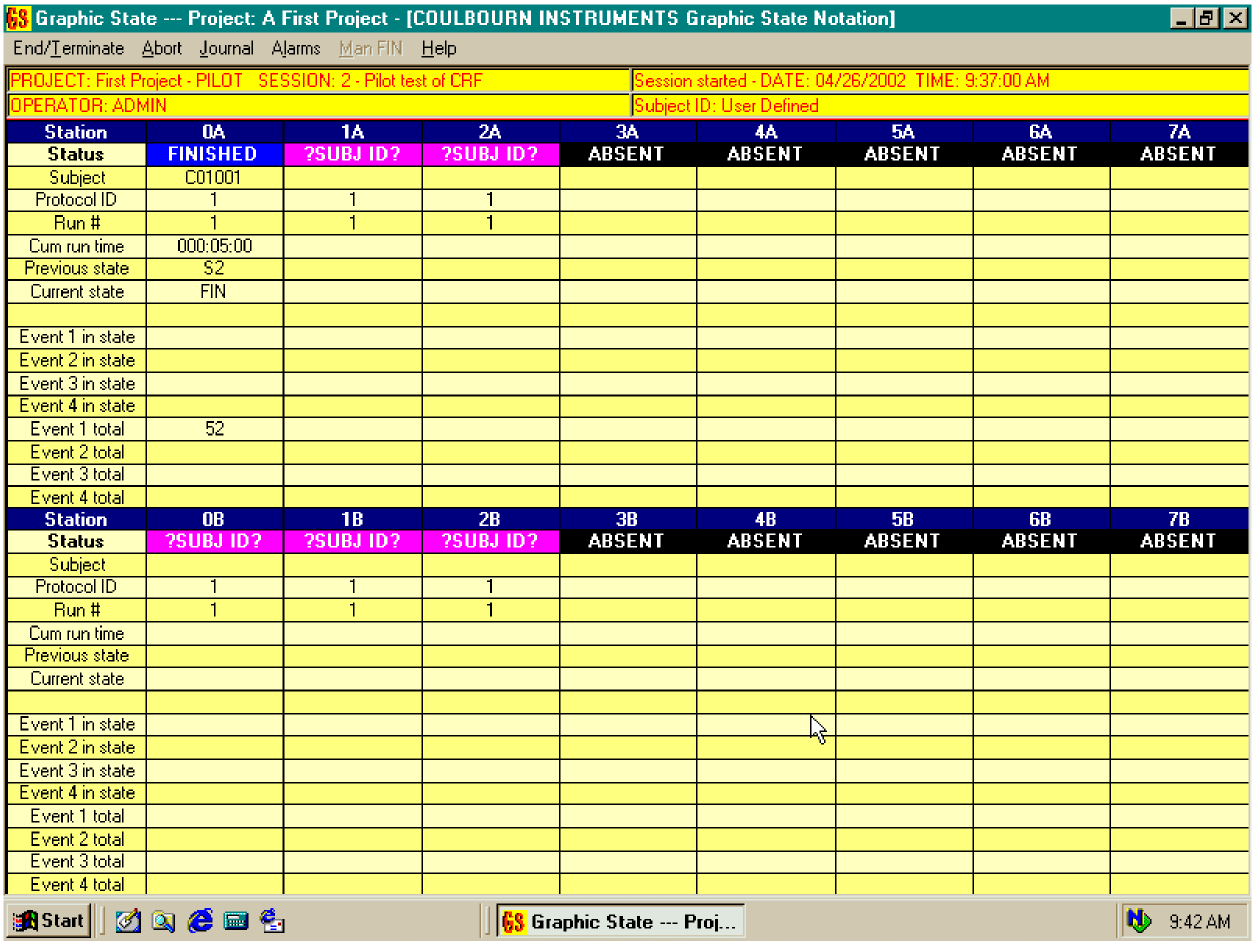
OPERATOR: ADMIN | Subject ID: User Defined

Station	0A	1A	2A	3A	4A	5A	6A	7A
Status	RUNNING	?SUBJ ID?	?SUBJ ID?	ABSENT	ABSENT	ABSENT	ABSENT	ABSENT
Subject	C01001							
Protocol ID	1	1	1					
Run #	1	1	1					
Cum run time	000:00:23							
Previous state	S1							
Current state	S2							
Event 1 in state								
Event 2 in state								
Event 3 in state								
Event 4 in state								
Event 1 total	4							
Event 2 total								
Event 3 total								
Event 4 total								

Station	0B	1B	2B	3B	4B	5B	6B	7B
Status	?SUBJ ID?	?SUBJ ID?	?SUBJ ID?	ABSENT	ABSENT	ABSENT	ABSENT	ABSENT
Subject								
Protocol ID	1	1	1					
Run #	1	1	1					
Cum run time								
Previous state								
Current state								
Event 1 in state								
Event 2 in state								
Event 3 in state								
Event 4 in state								
Event 1 total								
Event 2 total								
Event 3 total								
Event 4 total								

Start | Graphic State --- Proj... | 9:37 AM

When the run ends (after 5 minutes in my case), the header area for the station will show a blue FINISHED. We can see from the screen how many lever presses were recorded over the run (52 for my sample here).



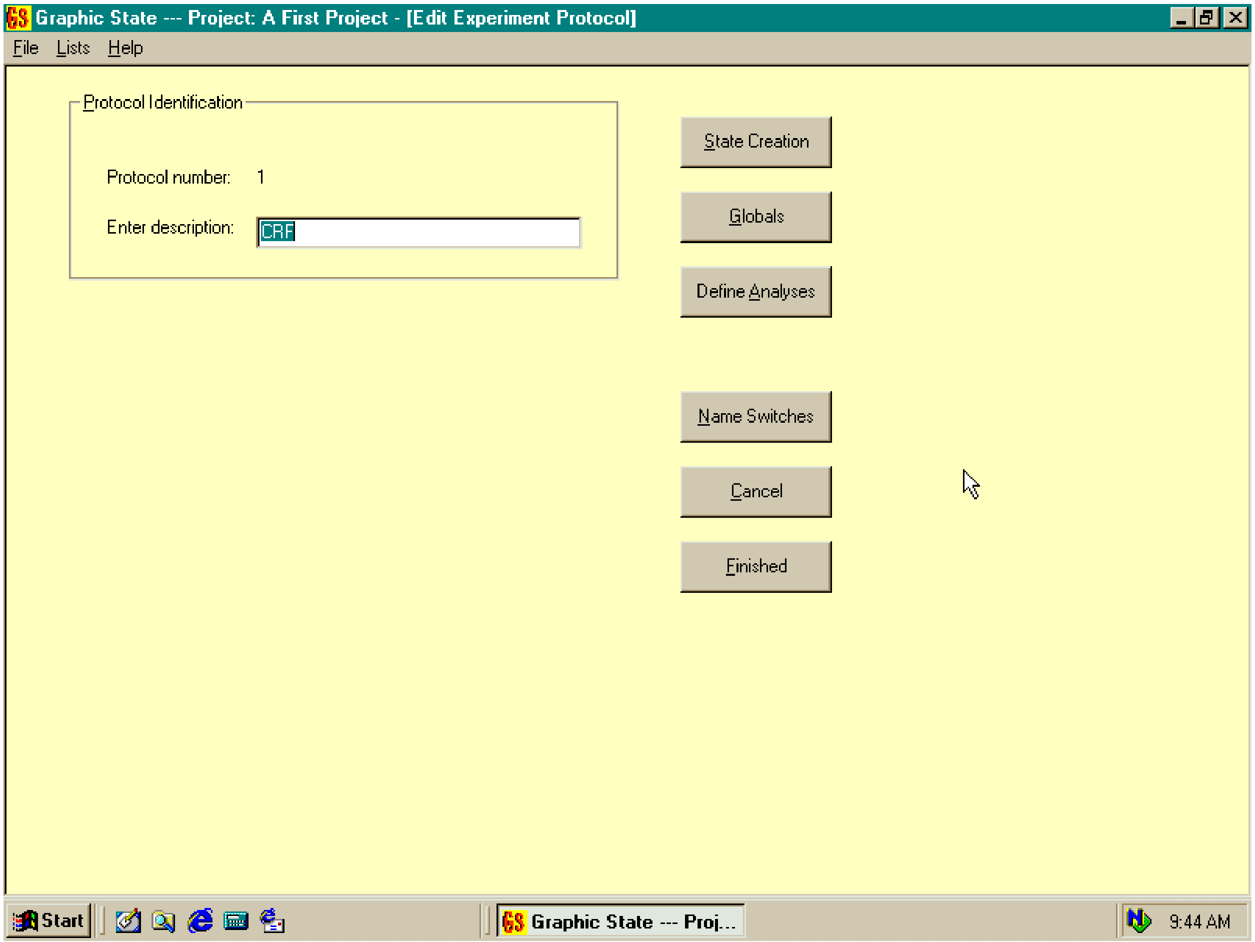
Select End/Terminate from the menu at the top of the run screen and you will be returned to the main menu screen.

You may be asking, "Why did I not get to perform data analysis at this point?" This is because you have not yet defined any analyses to be done. In a later example you will see what happens at the end of a run when analyses *have* already been defined.

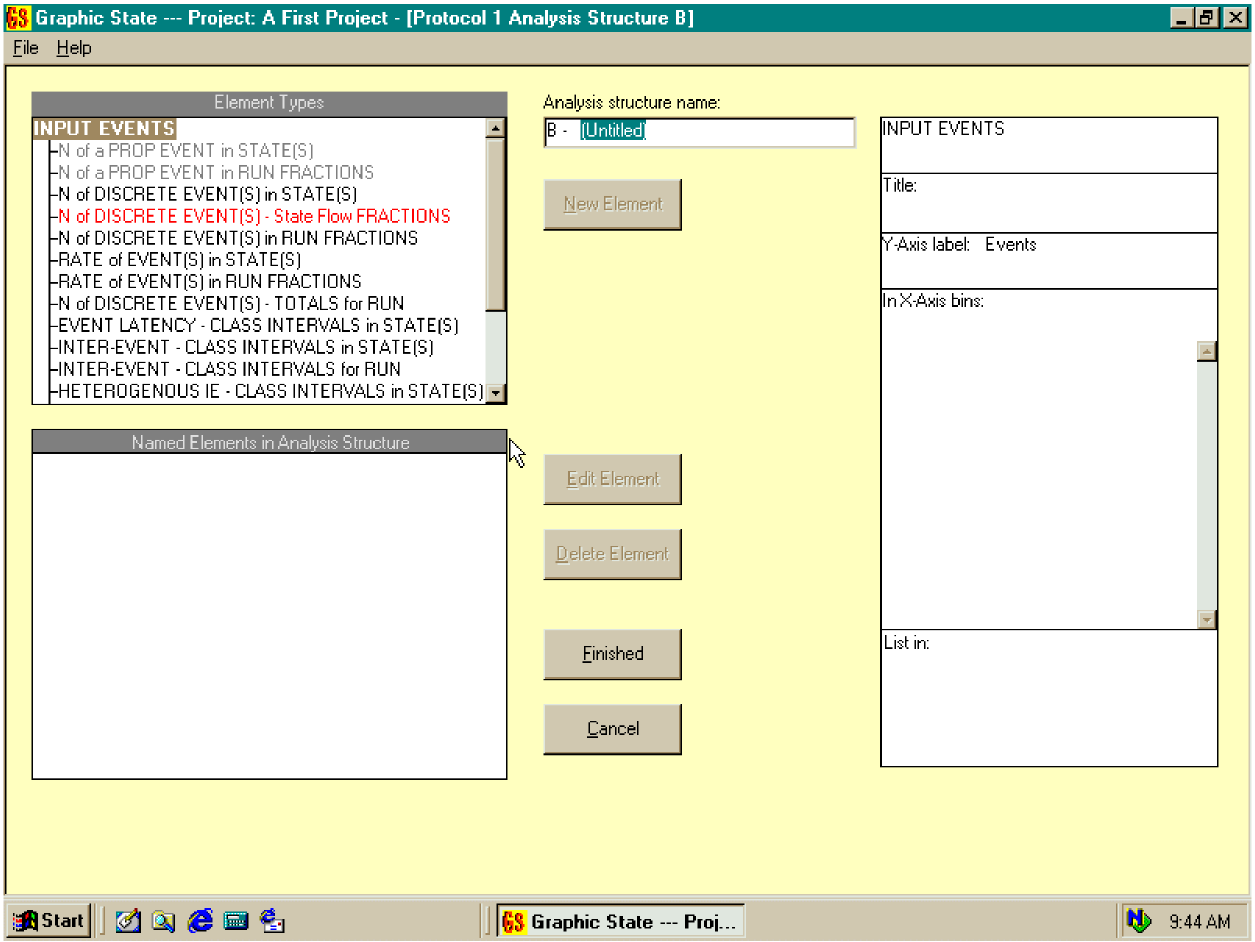
Analyzing your data

We can add analysis elements to the protocol now (or at any time in the future as well). From the File drop-down menu, select Edit/Review Experiment Protocol. Select the first protocol and click on the Review button (or the Edit button in this case since we have not yet run this protocol in "Archive" mode). We are going to add analysis specifications, but we are not going to edit (i.e., change) the *logic* of the protocol. Therefore we can use the Review mode here and, in fact, we *must* use the Review mode for protocols that have been run with real data. This may seem confusing now, but just take it on faith for the moment.

When you have the starting screen for the protocol displayed (as below), click on the "Define Analyses" button.



What will appear is a screen for selecting "analysis elements" which will allow you to enter parameters for producing graphs, lists of computed data, and/or exported data for importing into Excel.

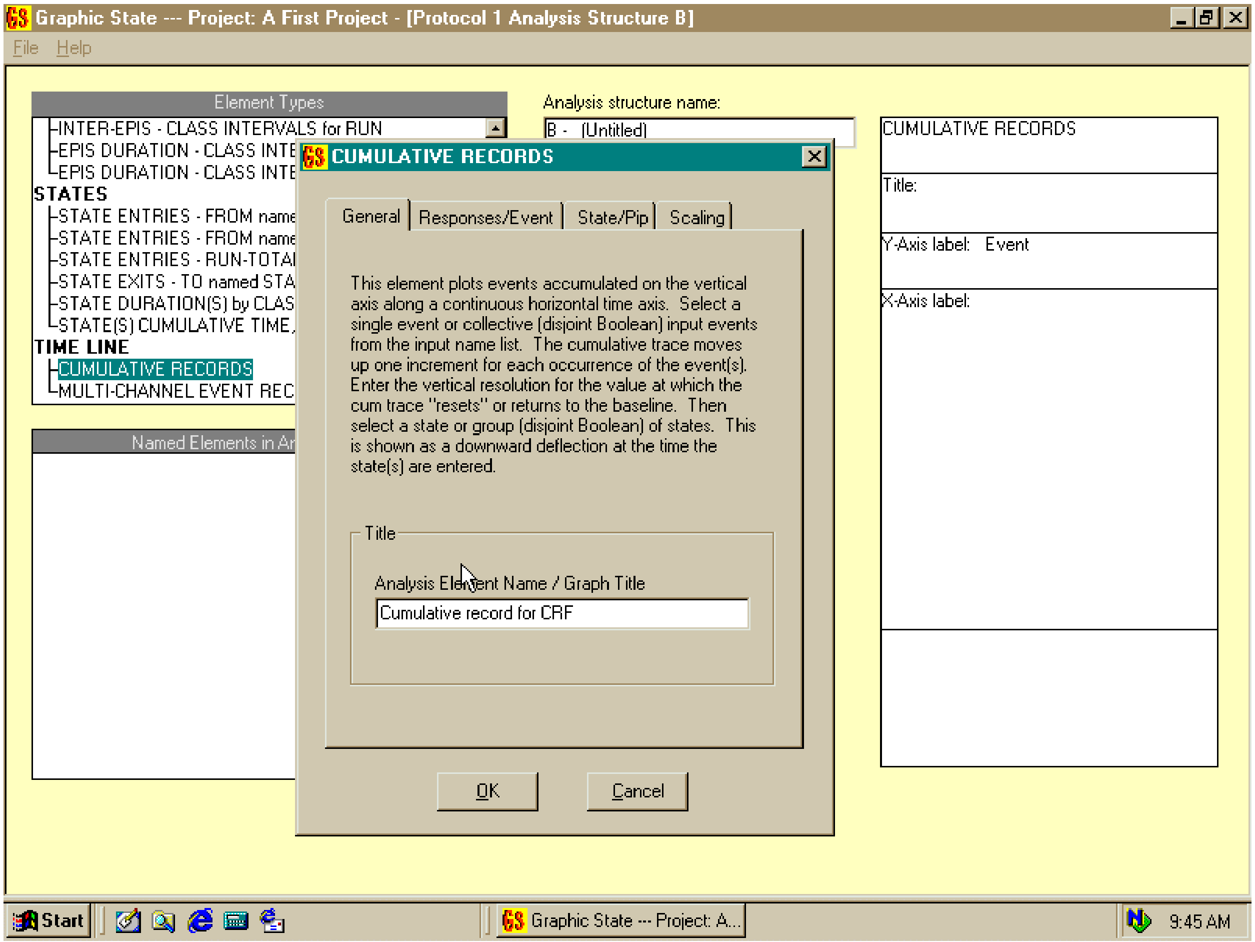


You can create one or more “Analysis structures” – groups of analysis elements. By default, the first such structure is named “B – Untitled.” You may leave the name as is, or you may overwrite the “Untitled” with any name of your choice. Let’s leave it as is for now.

The upper left portion of the screen contains a list of all of the possible analysis types. They are grouped logically into groups of elements that report 1) input events, 2) episodes of input events (e.g., meal analysis for feeding or drinking), 3) state entries, exits and durations, and 4) time lines (including cumulative records). The only way to really become familiar with all of the data analysis possibilities is to experiment with them. We will go over a few of the basic ones in this tutorial.

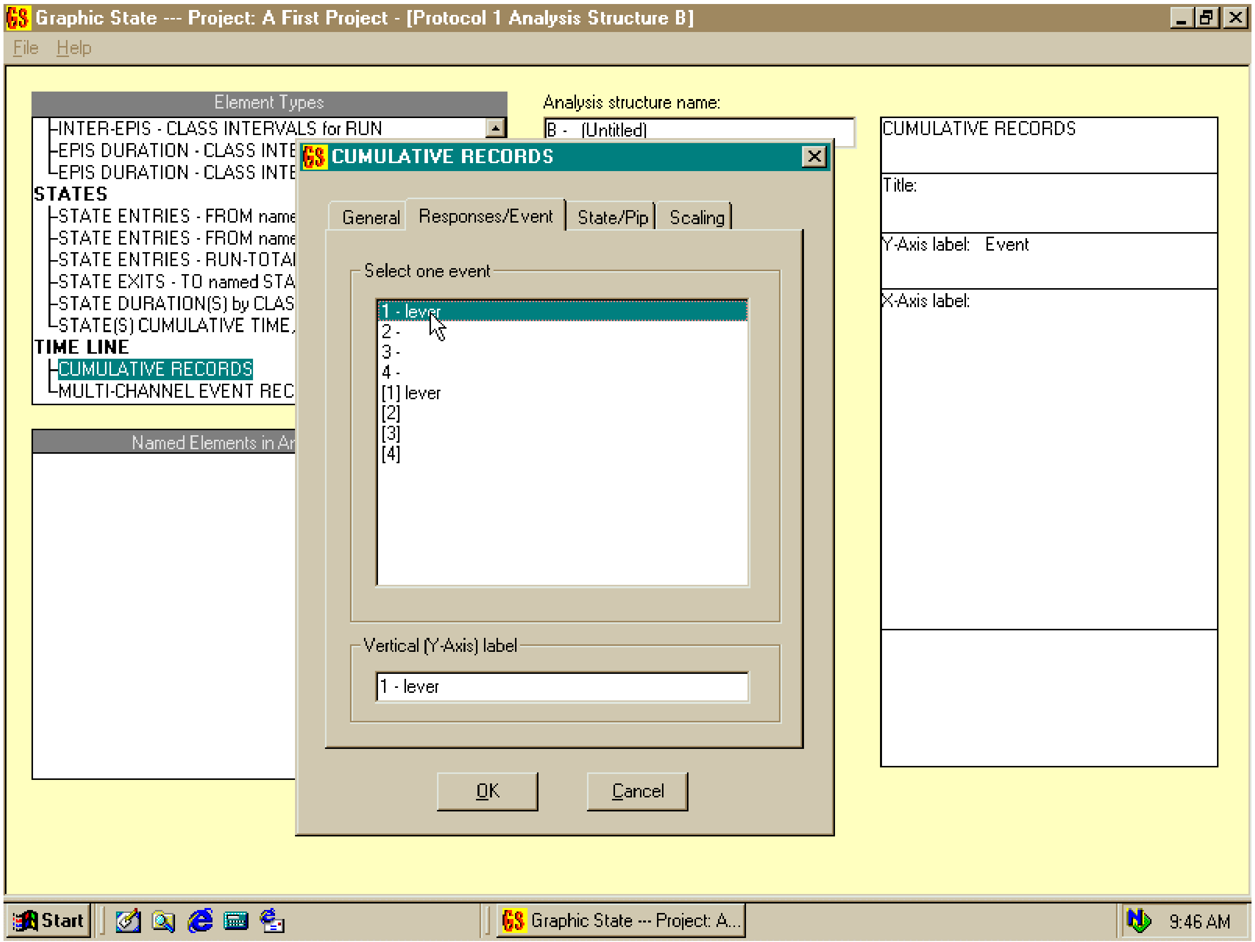
Let’s start with the most basic reporting tool of the operant experimenter – the cumulative record. Highlight the element (near the bottom of the list) called

“CUMULATIVE RECORDS” and then click on the “New Element” button. This will bring up a parameter screen with four tab folders.

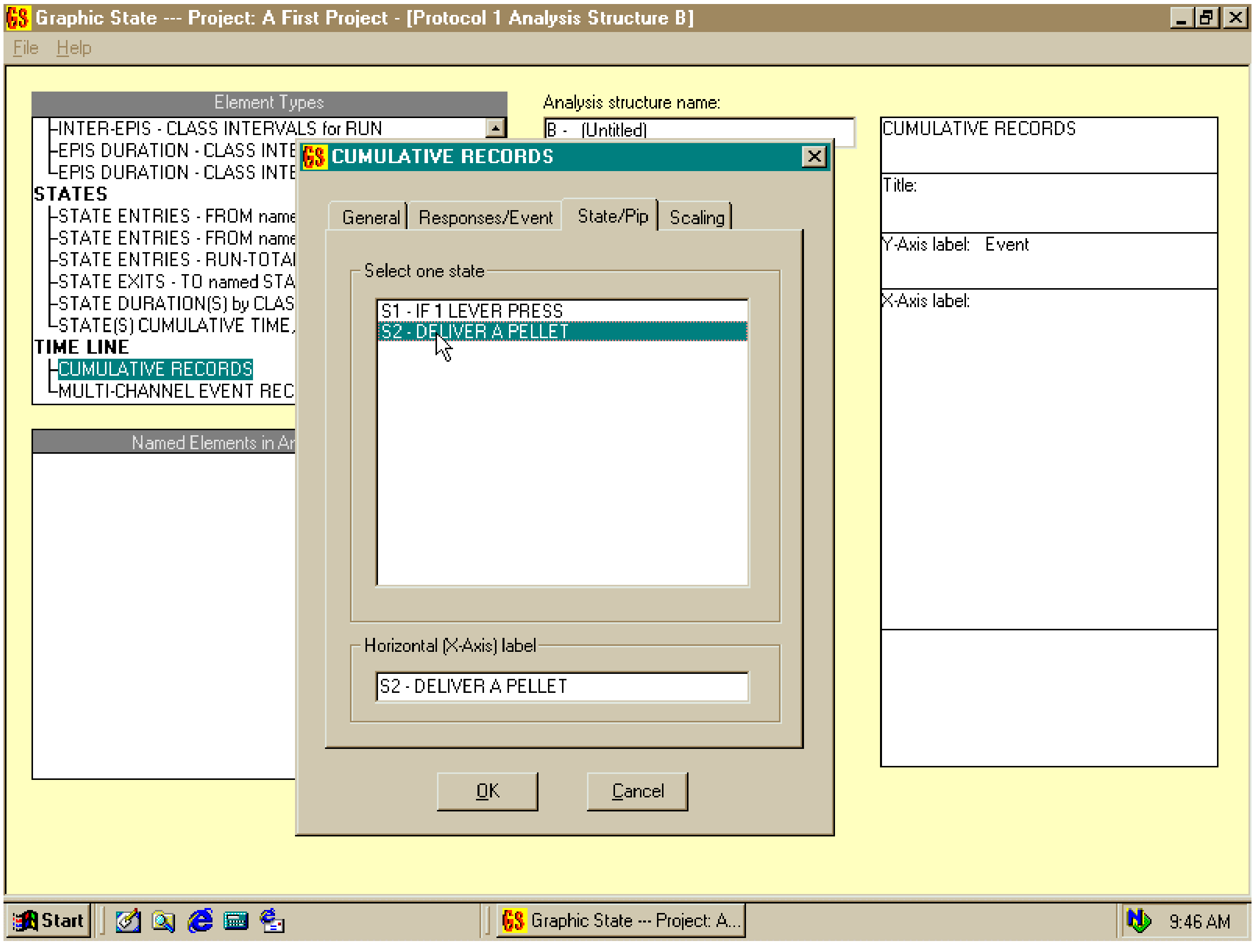


On the “General” screen that appears first, type in a title to appear on the graph. For our example, type in “Cumulative record for CRF.”

Next, click on the tab “Responses/Event” and, in the window that appears, select the event (i.e., the response switch) that you want to analyze. In our case it will be Switch 1, the lever.

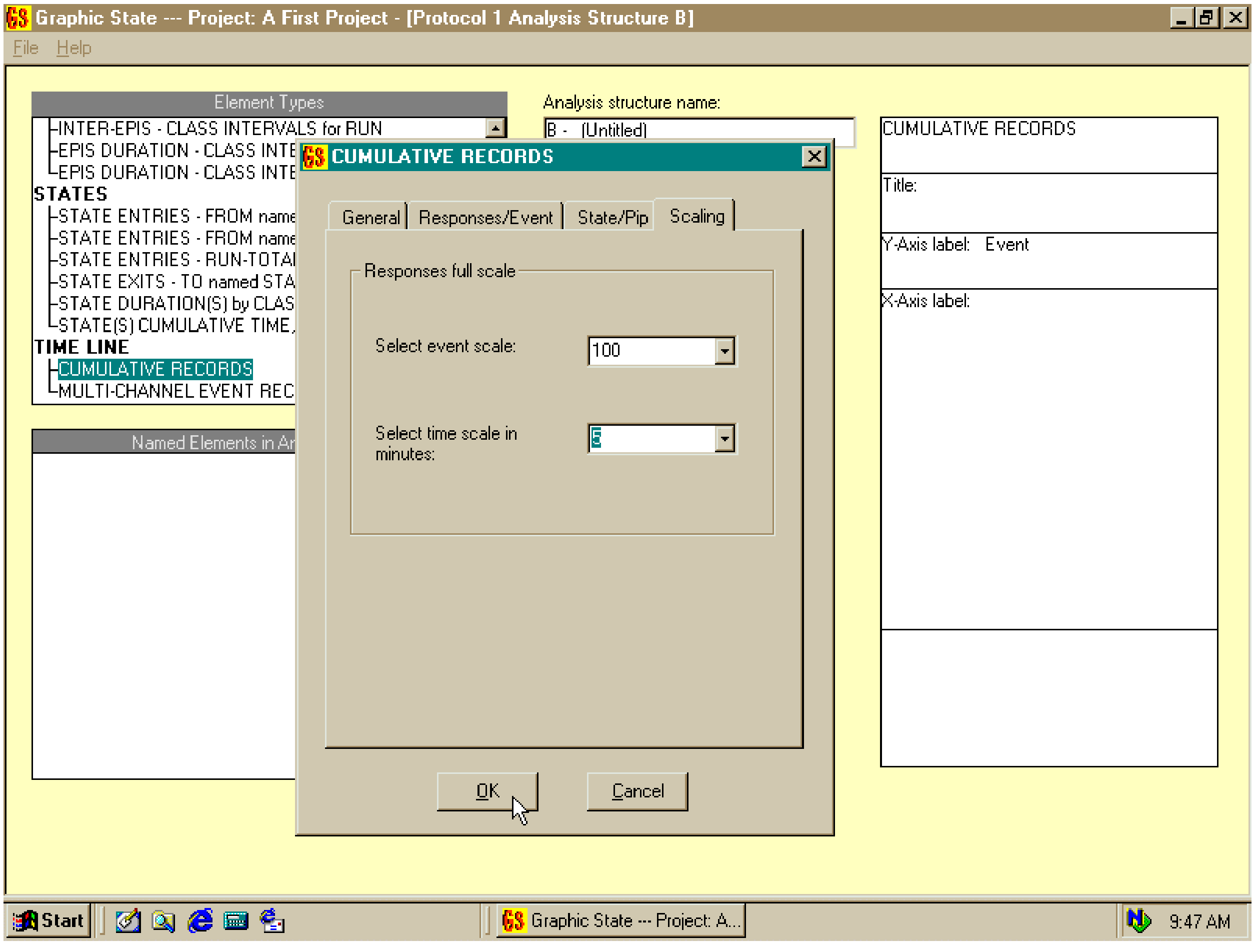


A cumulative record is a continuous trace that moves to the right across the page as time passes and moves up the page by one unit for each response. In addition, it traditionally includes a brief downward mark (called a “pip”) whenever a reinforcer is delivered. To specify the state in which the reinforcer is delivered, click on the tab labeled “State/Pip” and then select State 2 (“DELIVER A PELLET”).



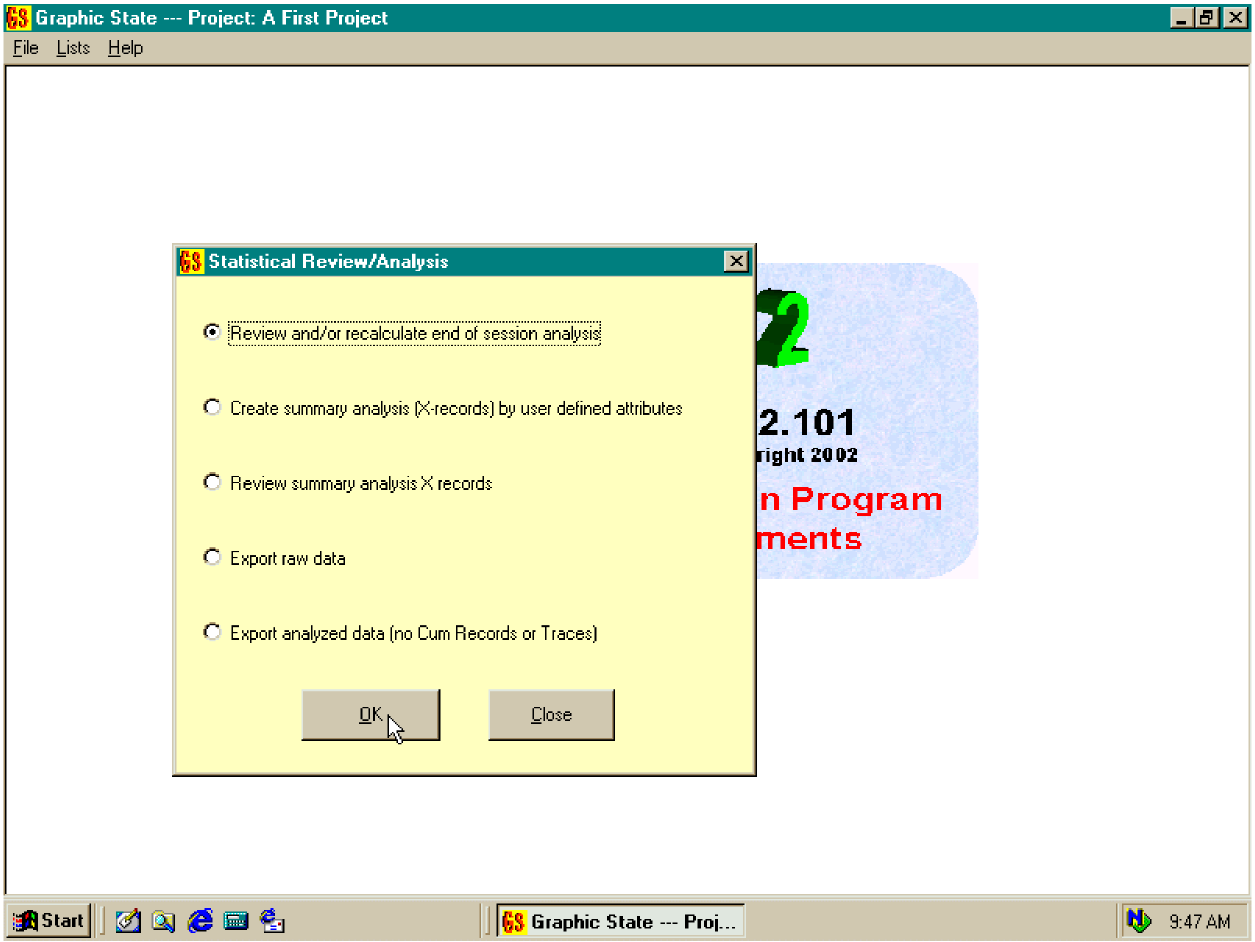
Finally, select the tab labeled "Scaling." Here you will select two scaling dimensions for the graph. The first is the maximum number of responses that should be recorded (by moving the trace vertically up the page) before the trace is "reset" to the bottom of the page again. Your choices are 100, 300, 500, 700, or 1000; for this example, select 100. The second parameter is the time period that should be displayed on each page of the trace. The choices range from 5 to 240 minutes. The resolution is better with a shorter time period, and also since our run was only 5 minutes long in total, select 5 minutes.

Finally, click OK to accept these parameters.

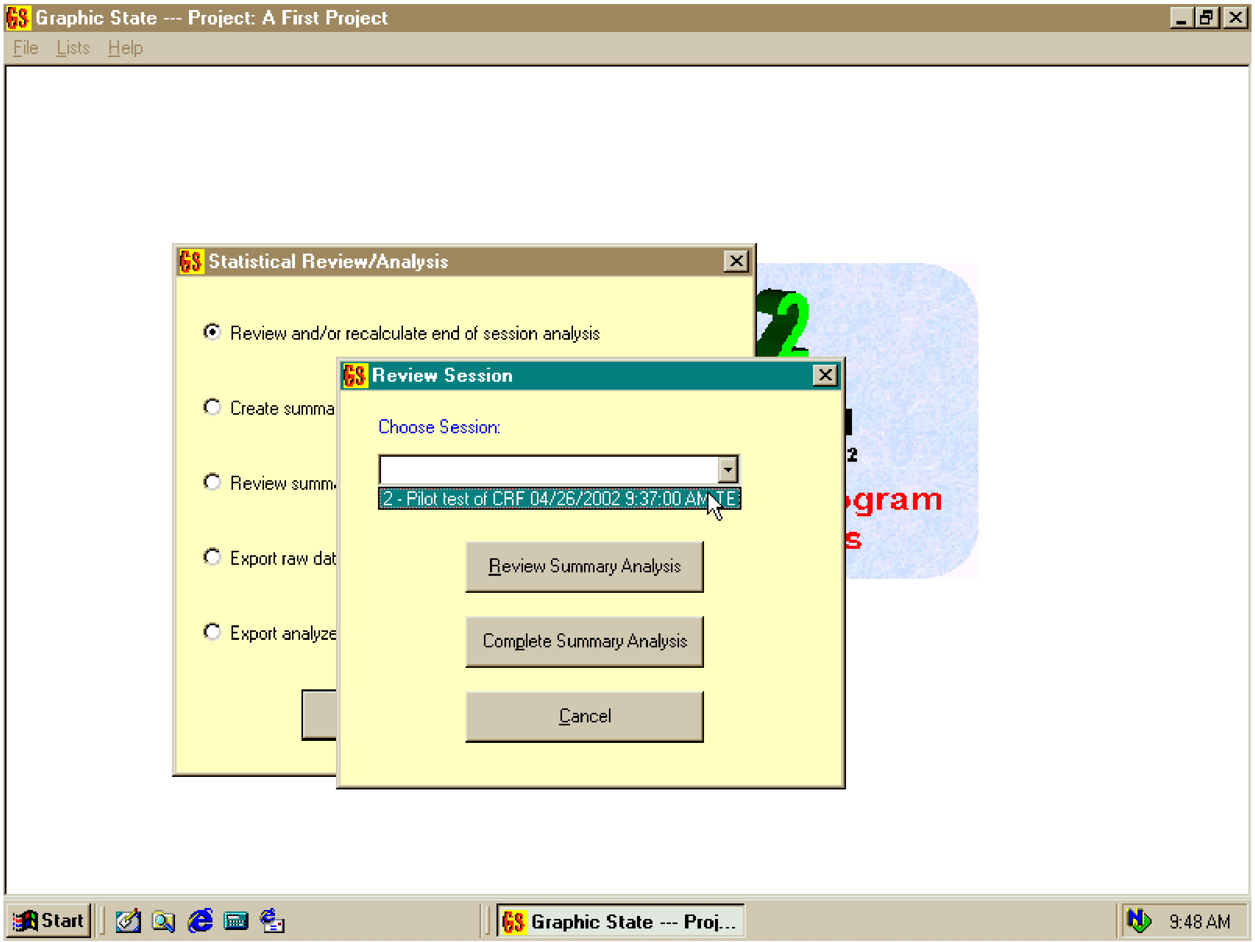


Click on the Finished button on the data analysis screen and then on the protocol definition screen to return to the main menu.

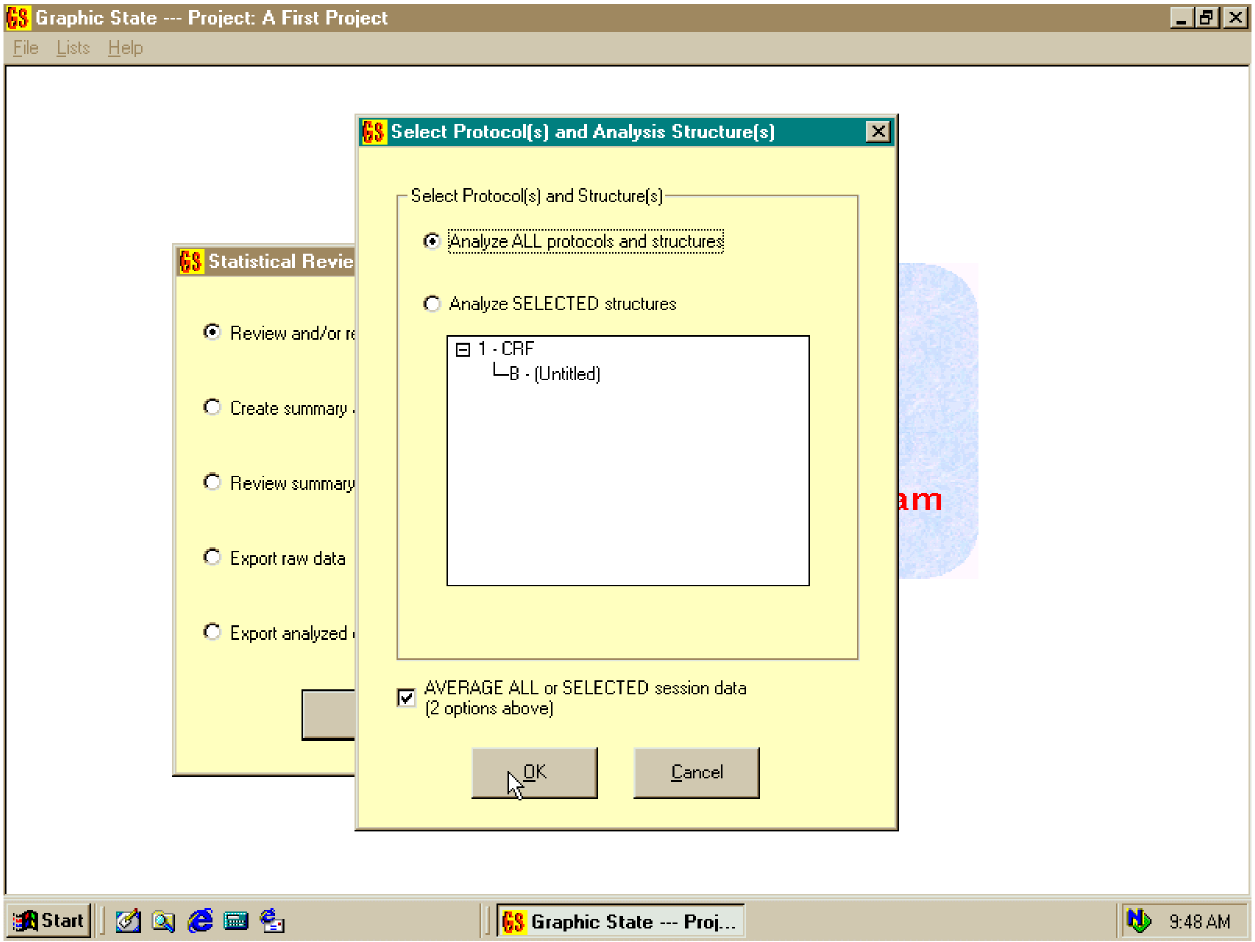
Now we can perform this analysis on the data we have already collected. Select "Statistical Review/Analysis" from the File pull-down menu. A screen with several analysis options will appear. Select the option "Review and/or recalculate end of session analysis" and click on OK.



Use the drop-down list in the "Choose Session" field to select the session to analyze. At this point there should be only one session listed there. In the future, there will be many. A sequential number, the description that you entered when you ran the session, the run date, and the starting time for the session, identifies the sessions. After you have highlighted the session, click on Complete Summary Analysis (because we have not yet run the analysis). If you come back here again in the future to look at this session, you will select Review Summary Analysis.



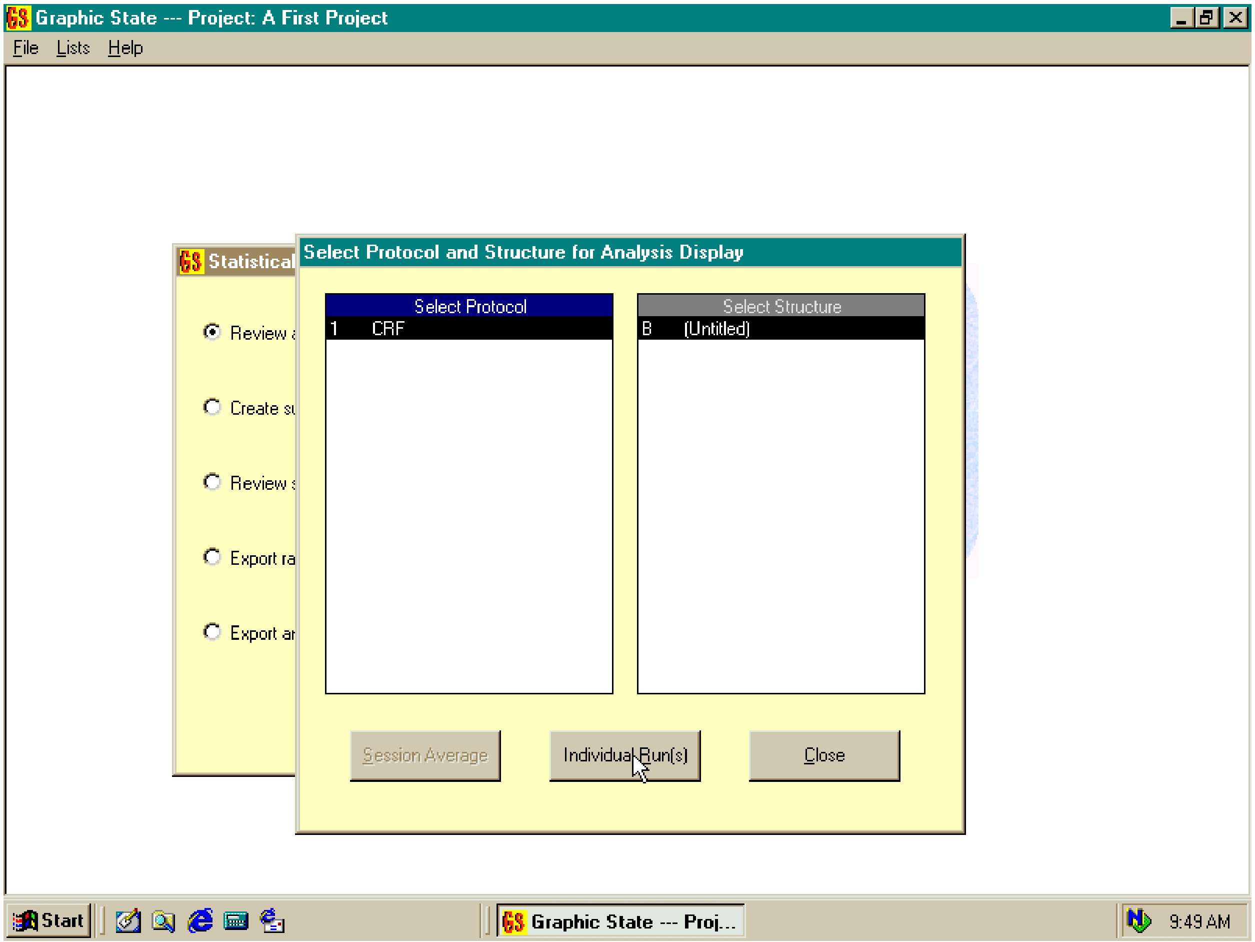
The screen that comes up next will list all of the protocols that were run in this session (only one, 1 – CRF, this time) and all of the Analysis Structures defined for each protocol (again only one, B – Untitled).



We have the choice of analyzing all protocols and structures or of selecting particular ones. Leave the option selected to analyze them all. We also have the option of computing averages across the session (the check box near the bottom of the screen). In our case, we only ran one animal in one station, so averages are meaningless. In this case, it does not matter whether the box is checked or not. Click on OK to proceed.

After a seconds you will get a message that the analysis is complete. Click OK here also.

Now you will see a screen that lists each analysis structure that is available within each protocol used in the session. In our example, only one protocol was run and it has only one analysis structure, and they are already highlighted. You would have the choice of looking at session averages or individual runs, but since there was only one run, the averages option is grayed out.

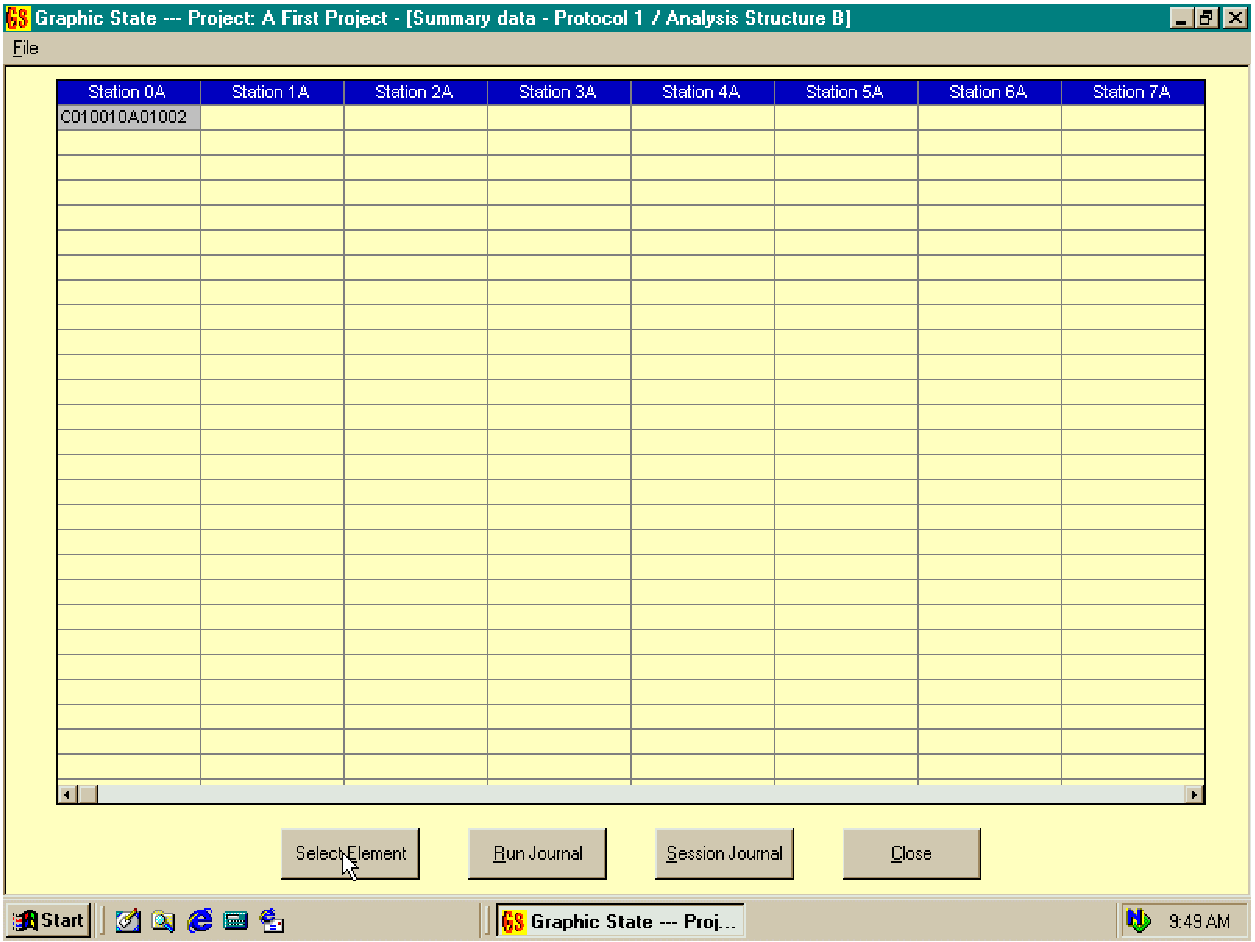


Click on Individual Runs to continue.

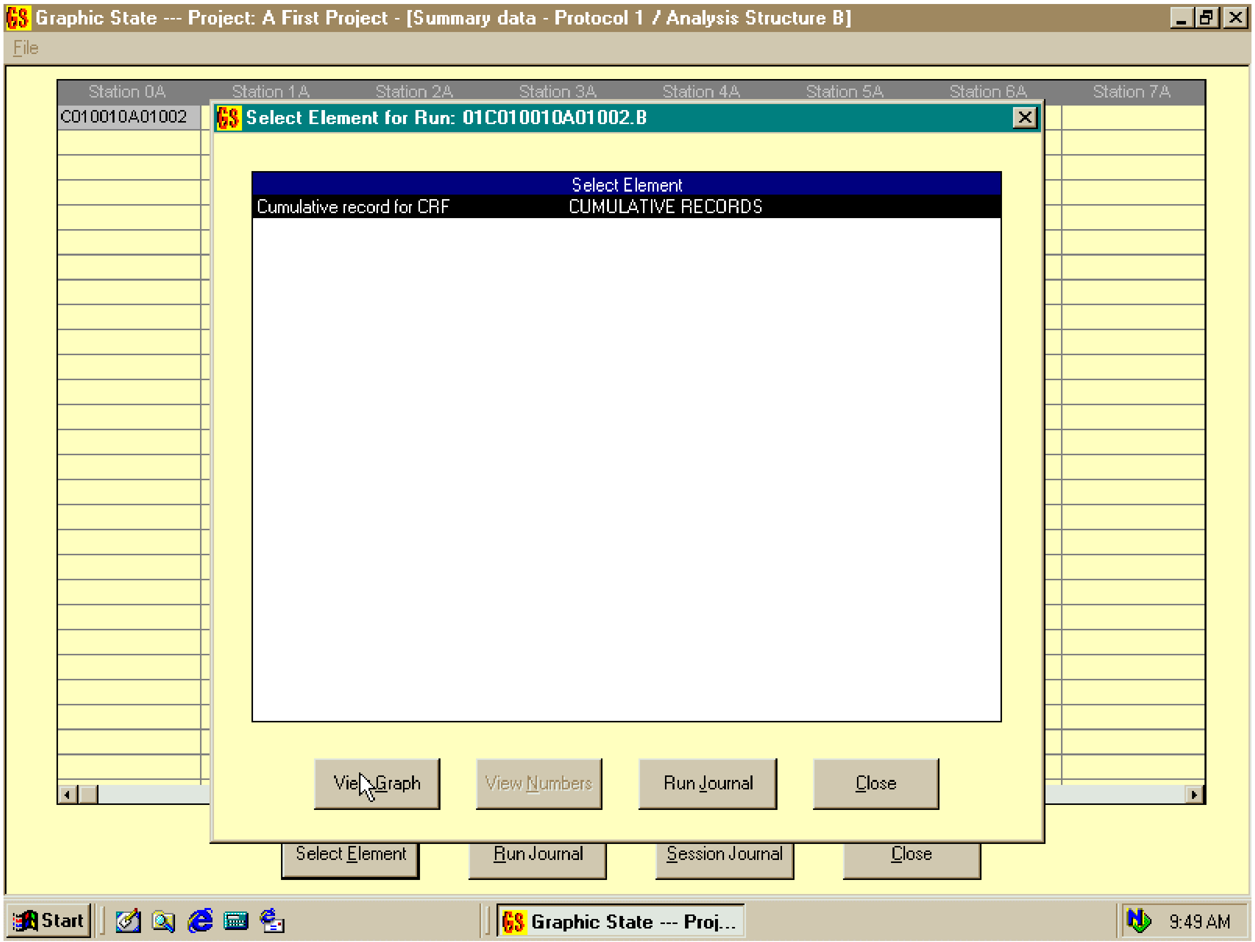
A grid that again looks like a spreadsheet is displayed. It will contain every run, in every station, that was part of this session. The columns are labeled with the Station IDs (0A, 1A, etc.). When there are more than 8 possible stations (as in this case where one half of a Linc is used for each station), the second group of 8 is shown on a second page; use the scroll bar at the bottom of the screen to see them.

Under each station heading, each separate run is listed. It is identified by a string that is made up of the Animal ID, the Station ID, the Run Number (within a session) and the Session Number. In our example, we ran animal “C01001” in station “0A”, as the first run (“01”) in this station for session “002”.

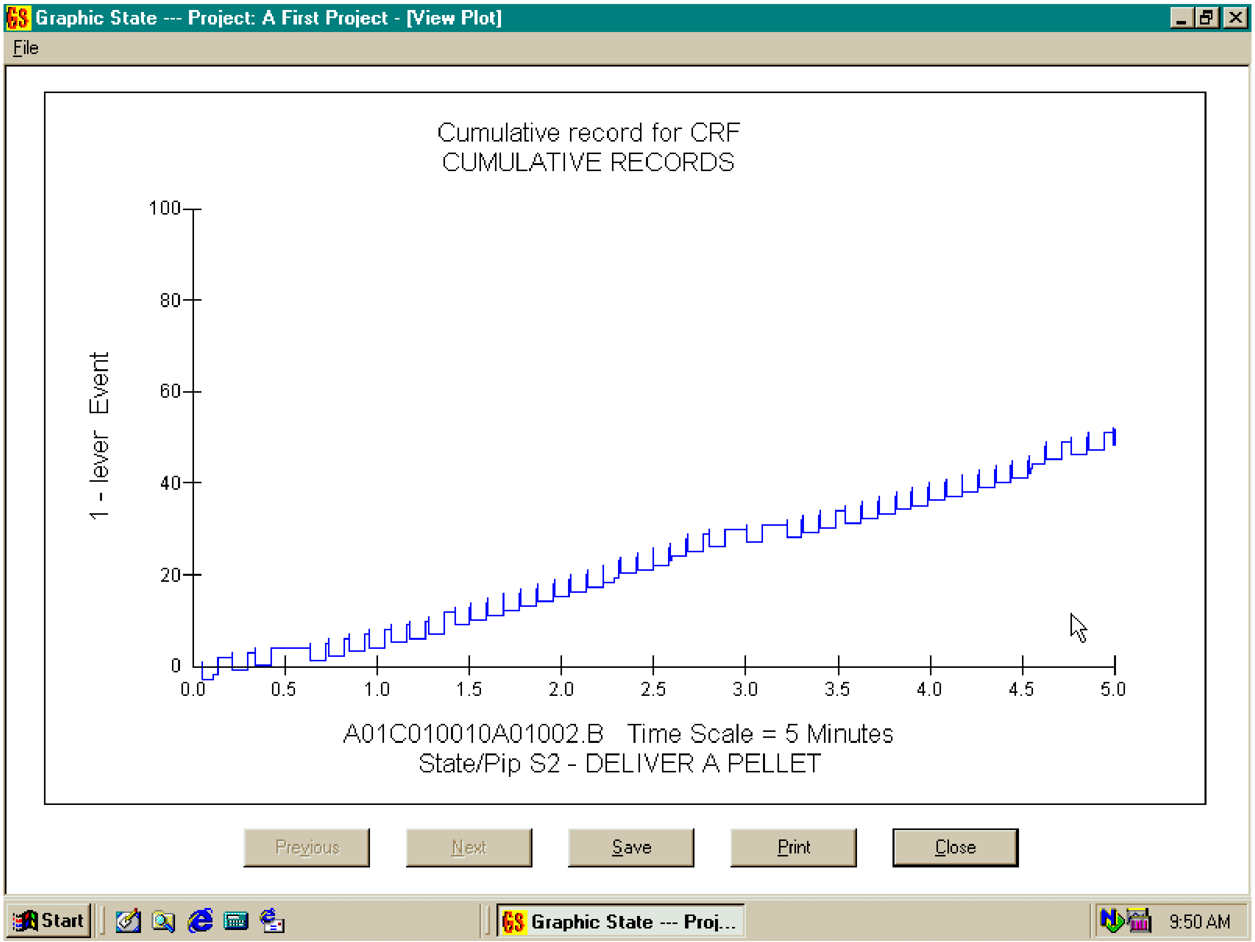
.



Highlight the grid "element" for the run you want to view. In this case, it is already highlighted since it is the only run. Click on "Select Element".



Each analysis element that we have defined for this protocol will appear in a list. When we highlight an element, we will ordinarily have the option of viewing a graph or of viewing the data (as numbers) in a list. We will look at an example of the numbers option later, but for a cumulative record, the graph is the only option. Click on "View Graph."



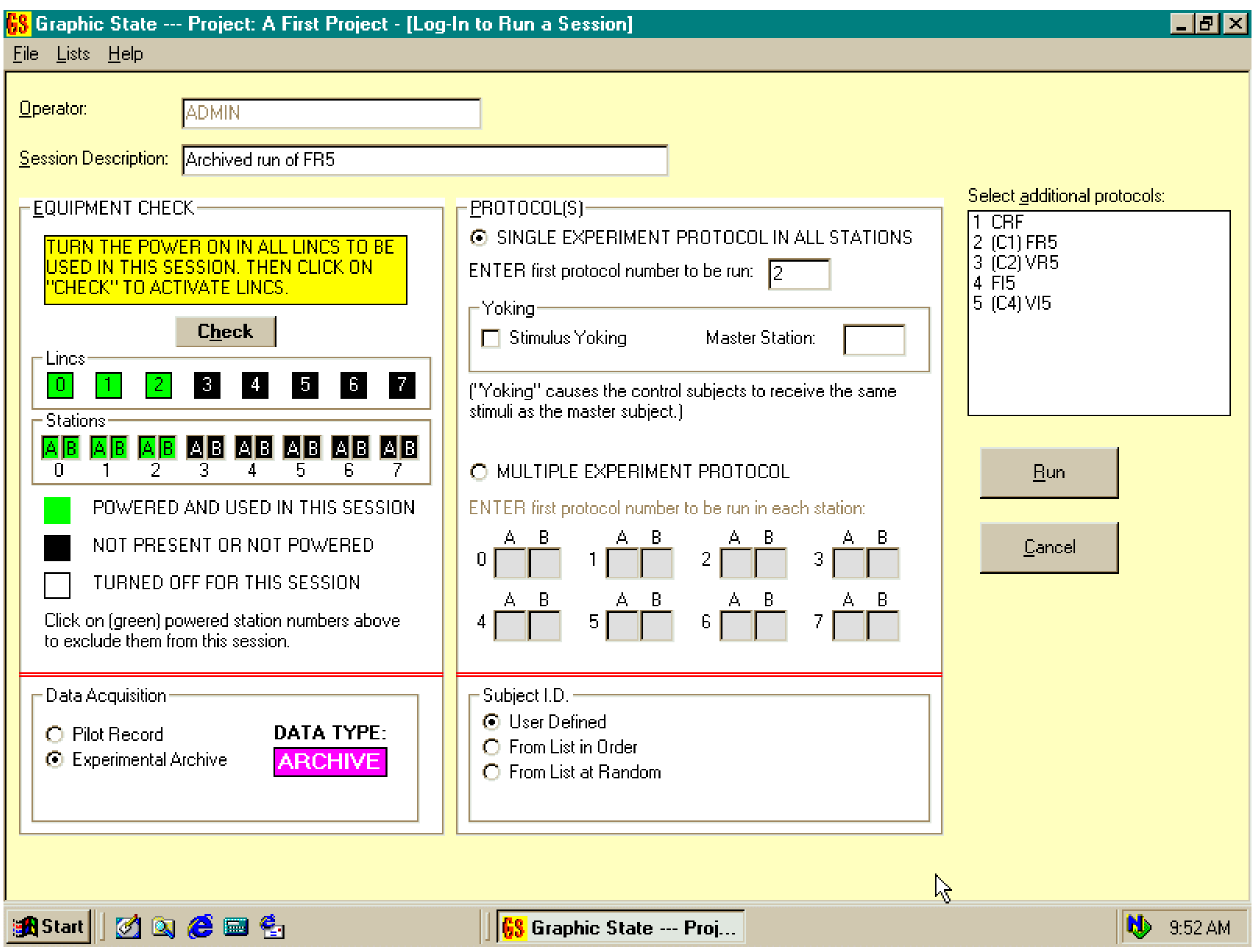
This is an example of a cumulative record for a five minute run. The downward "pips" are somewhat exaggerated in this example graph because the time frame of the graph is short and because the pip lines stay down for the entire duration of State 2 (5 seconds). Nevertheless, you can see the upward movement as responses occurred. We will see more traditional looking cumulative records in later examples.

At this point, we have the options of printing the graph, saving it as a bitmap image file, or of closing this display.

Adding more analysis elements and creating some more data

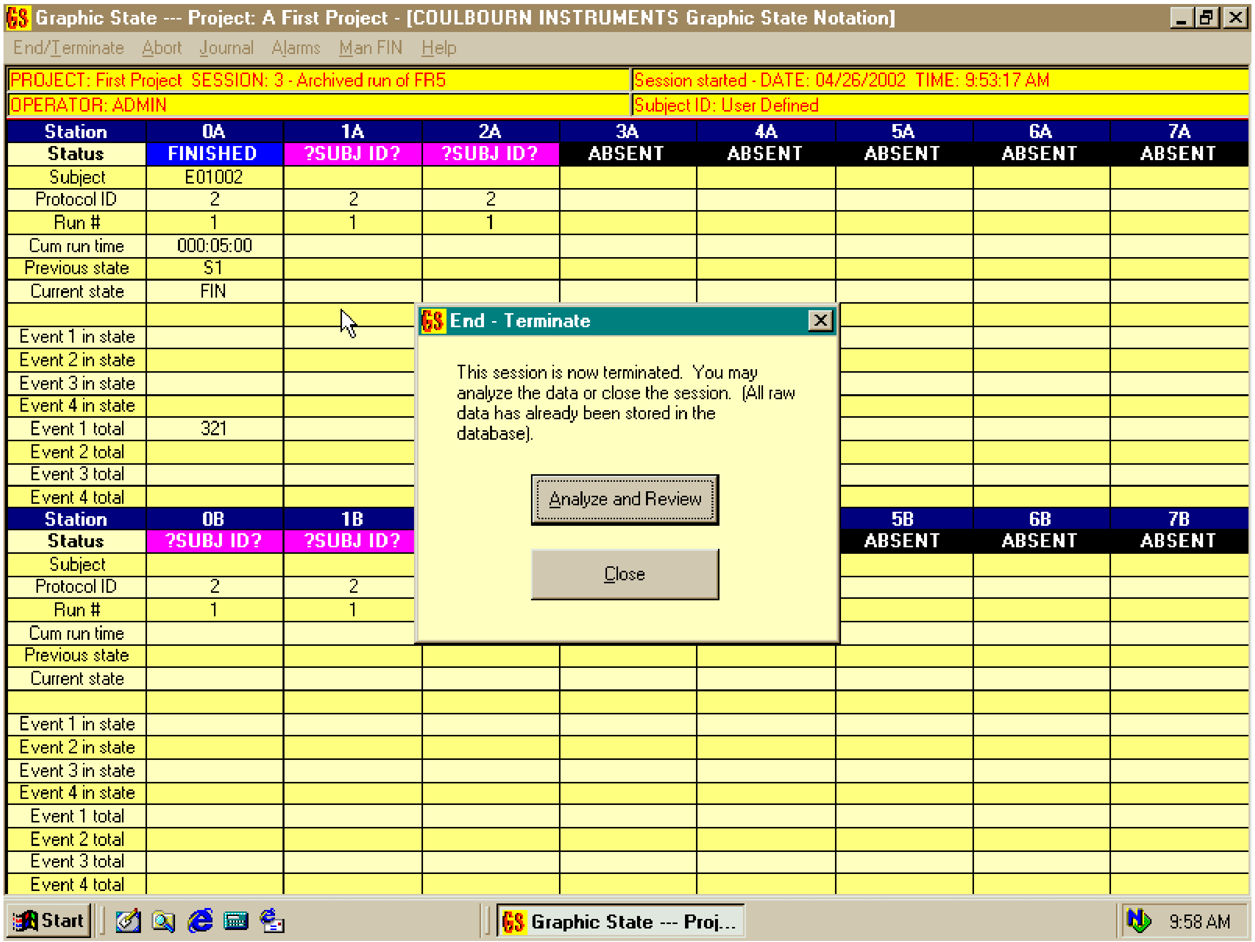
We have four other protocols defined, none of them with analysis elements. Select Edit/Review Experiment Protocol from the File menu, select the FR5 protocol, click on Review and then Define Analyses. Add an analysis element to produce a cumulative record. Repeat this process for the VR5, FI5 and VI5 protocols.

Now, prepare to pretend to be a rat (or use a real rat), and select RUN A SESSION from the File menu. Enter a session description and specify protocol number 2 to be run. This time, run with the “Experimental Archive” mode selected instead of “Pilot Record”. Click the Run button.

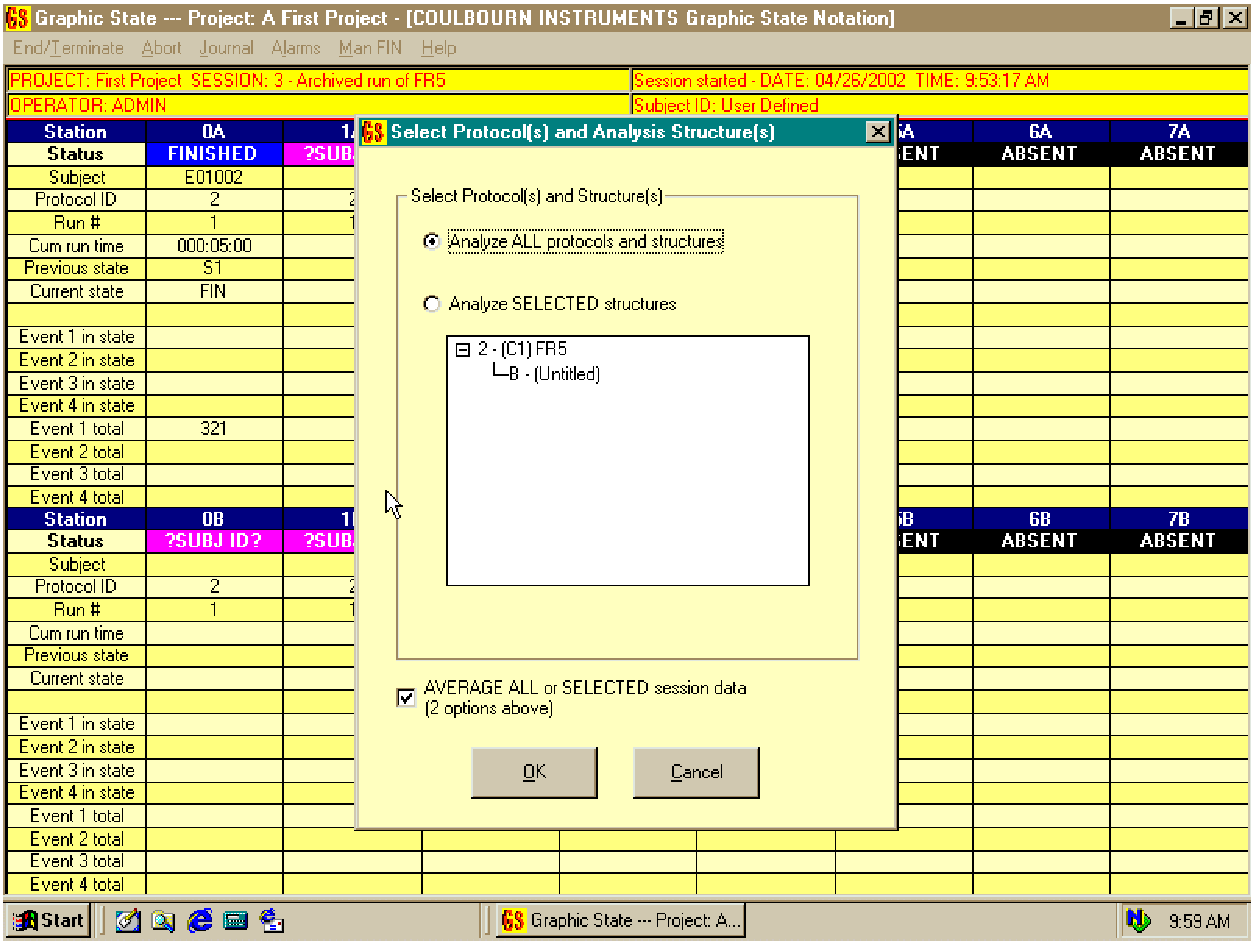


Enter a subject ID (I used “E01002” to identify animal # 2 in experimental group 1) and start the run. When the run is finished, click on End/Terminate at the top of the screen.

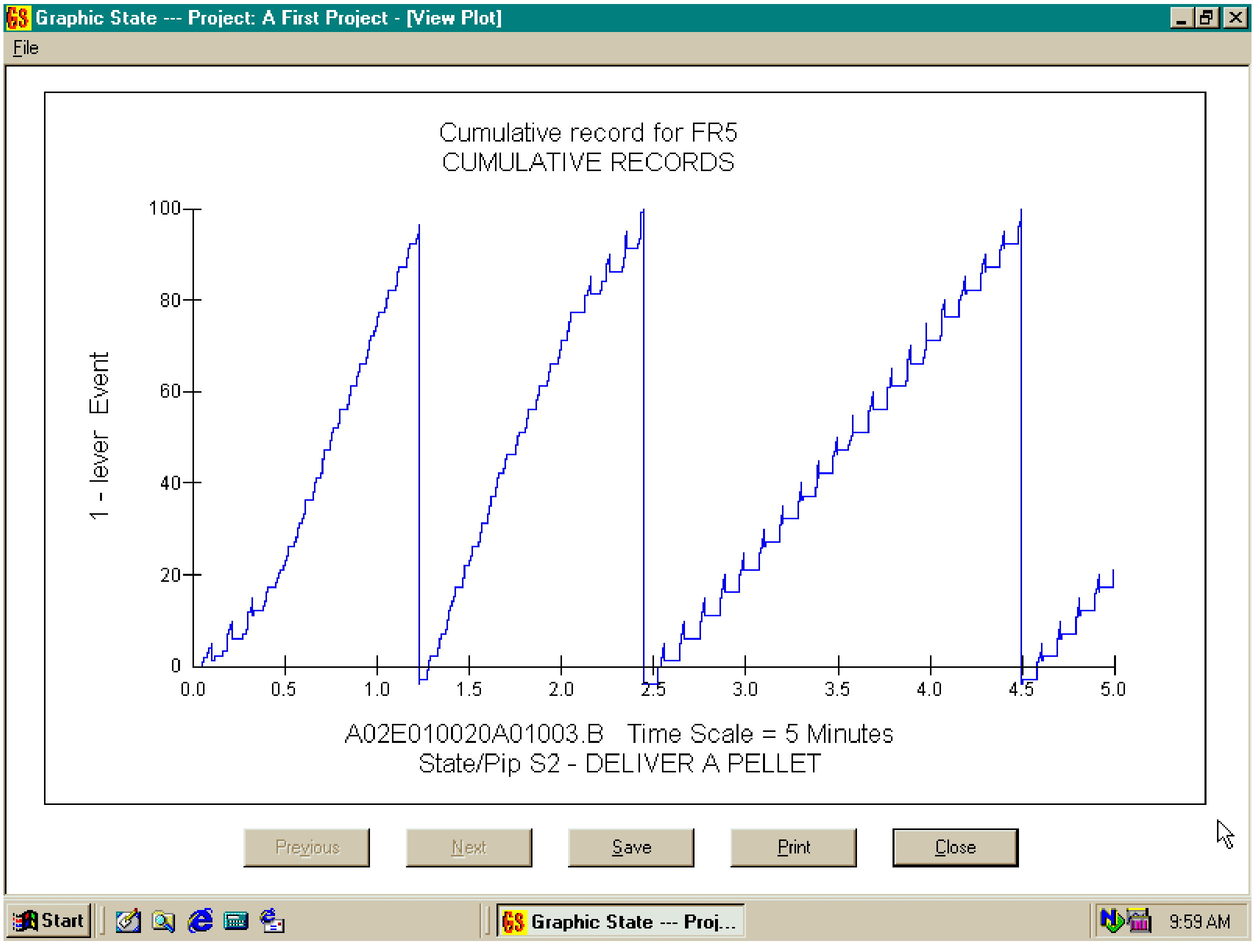
This time, because at least one analysis element has been defined, you will be given the opportunity to analyze data immediately.



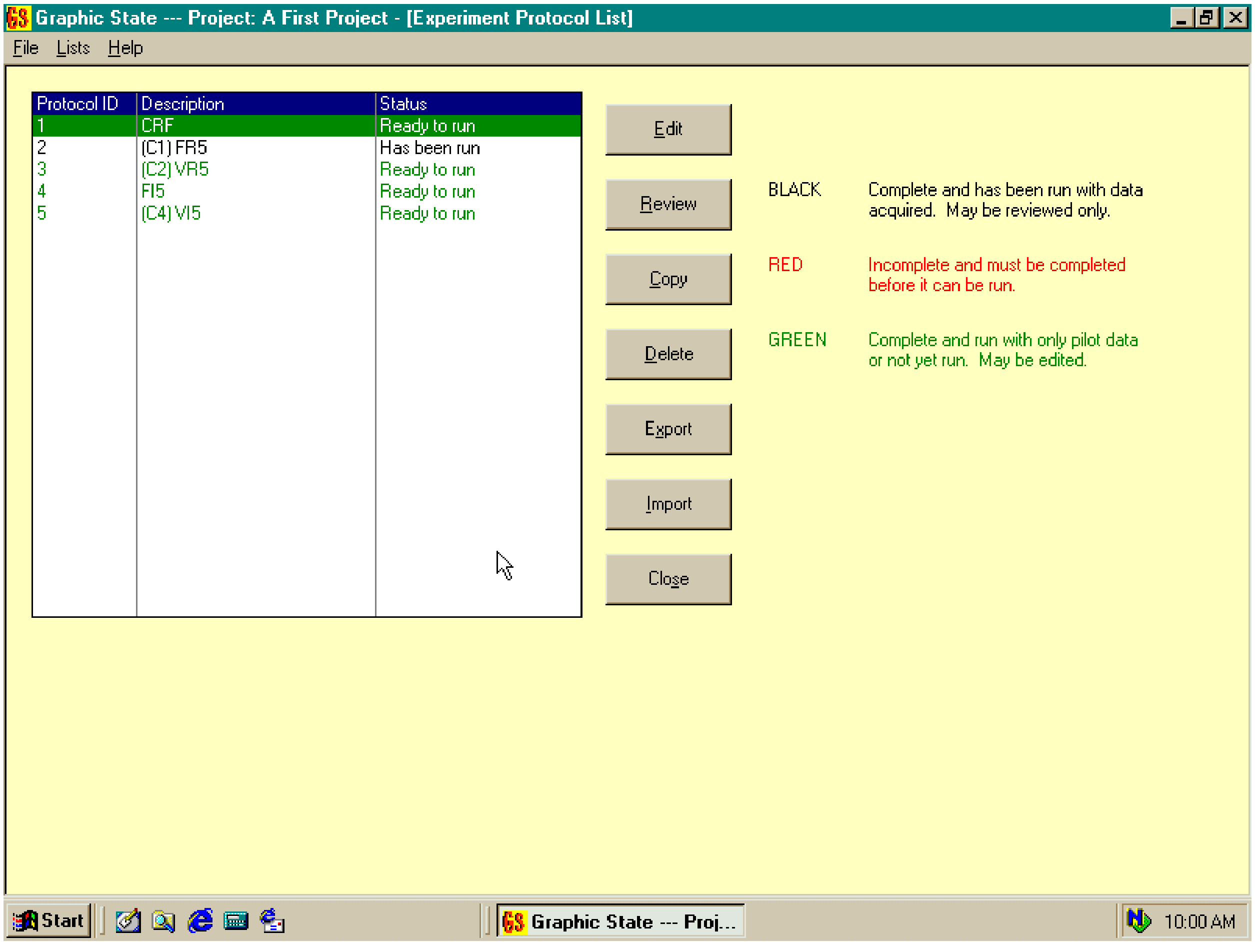
Click on Analyze and Review and you will start the series of data analysis screens.


Graphic State --- Project: A First Project - [COULBOURN INSTRUMENTS Graphic State Notation]
End/Terminate
Abort
Journal
Alarms
Man FIN
Help
PROJECT: First Project SESSION: 3 - Archived run of FR5
Session started - DATE: 04/26/2002 TIME: 9:53:17 AM
OPERATOR: ADMIN
Subject ID: User Defined
Station
Status
Subject
Protocol ID
Run #
Cum run time
Previous state
Current state
Event 1 in state
Event 2 in state
Event 3 in state
Event 4 in state
Event 1 total
Event 2 total
Event 3 total
Event 4 total
0A
FINISHED
E01002
000:05:00
S1
FIN
321
6A
ABSENT
7A
ABSENT
0B
?SUBJ ID?
6B
ABSENT
7B
ABSENT
Select Protocol(s) and Analysis Structure(s)
Select Protocol(s) and Structure(s)
Analyze ALL protocols and structures
Analyze SELECTED structures
2 - (C1) FR5
B - (Untitled)
AVERAGE ALL or SELECTED session data
(2 options above)
OK
Cancel
Start
Graphic State --- Project: A...
9:59 AM

This is how my cumulative record looked.



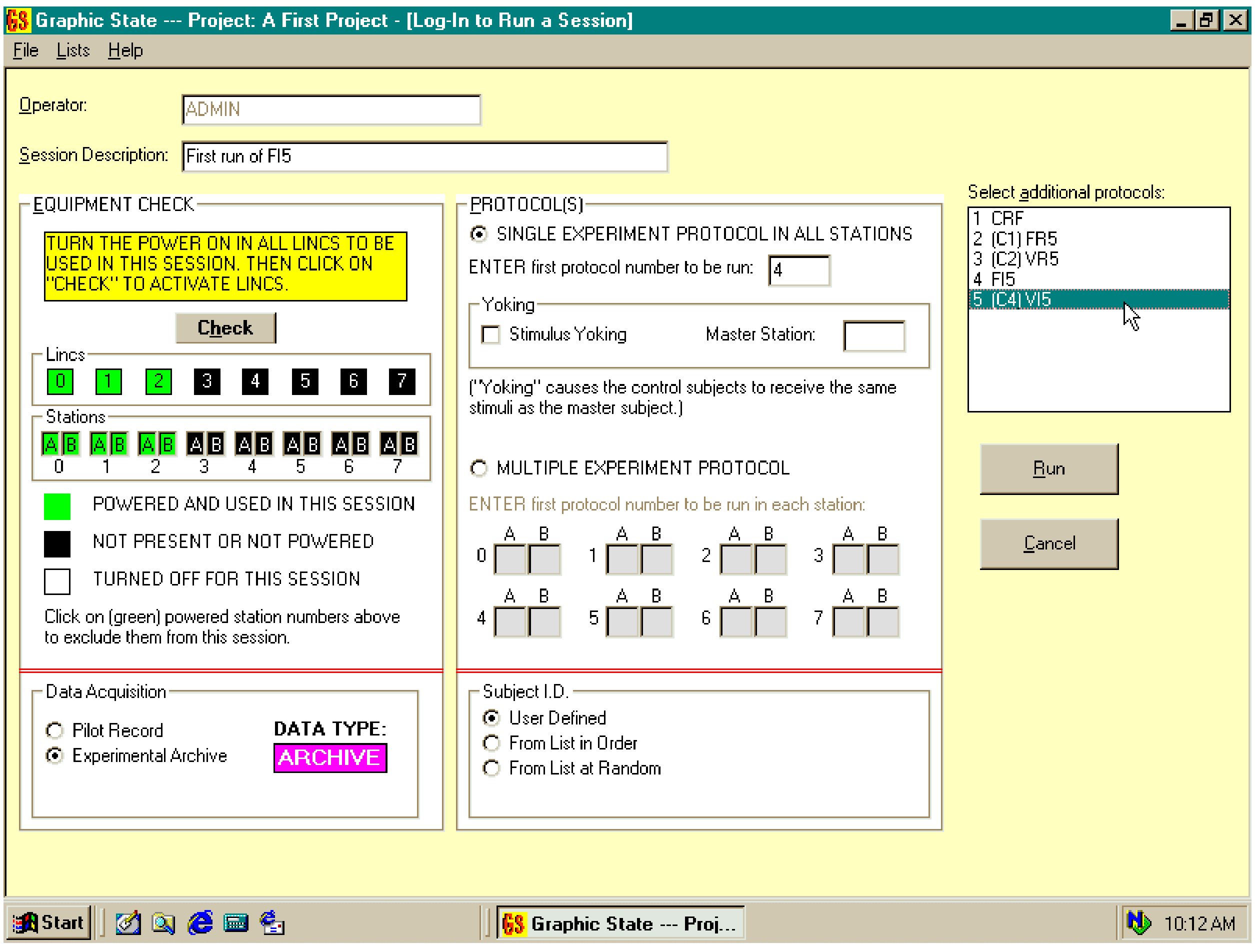
If you return to the list of protocols, you will see that now Protocol 2 is listed in black with the status “Has been run.” We have not yet run 3, 4 or 5 and so they are still listed in green. We have run Protocol 1, but only in Pilot mode, so it remains green also.



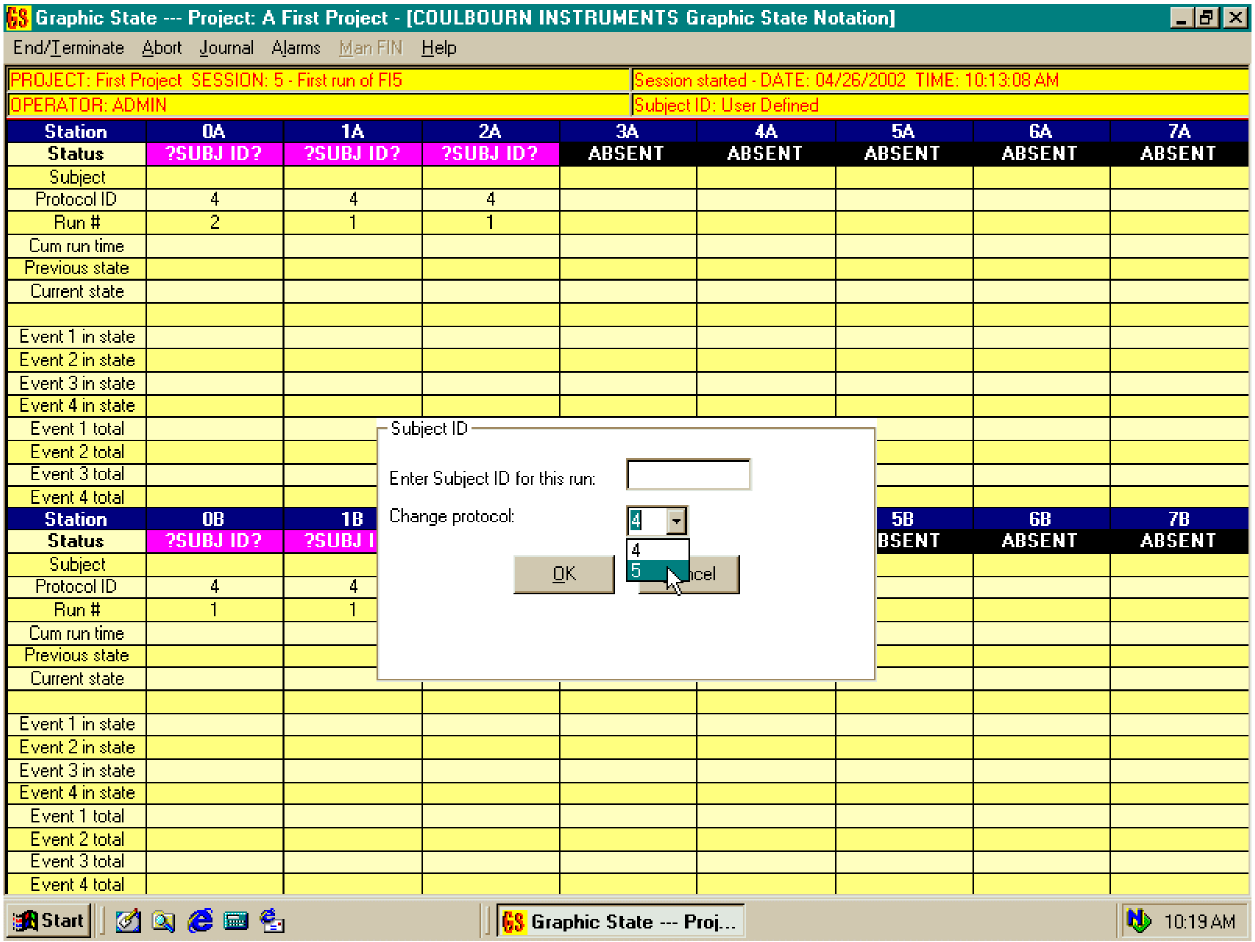
Add a cumulative analysis element to the remaining protocols (3, 4 and 5). Make a run of Protocol 3, the VR5. To make realistic looking data, when I am "playing rat", I press the lever repeatedly until the feeder light comes on on the front of the Linc. Then I count to three (pretending that I am retrieving a pellet and eating it) and then go back to the lever. Sometimes I am a little slower than other times. Sometimes I overrun with the lever (press it more times than necessary). All of this simulates what happens with a real rat after it has been trained.

Next, let's run Protocol 4 and then Protocol 5 in the same session sequentially in the same station. Select RUN A SESSION from the File menu. Do the station check (with the Check button) and enter a description as usual. This time, notice the list on the upper right part of the screen that is labeled "Select additional protocols." If you are planning to change the protocol running in a station sometime during the session, highlight the additional protocols (i.e., other than the first one, which is still entered in the field “ENTER first protocol to be run”) that you wish to use. The program will load all of these protocols into memory before the session begins to avoid slowing down for multiple disk reads when the new protocol is needed.

For our example, enter “4” as the first protocol to be run and highlight number 5 in the list.



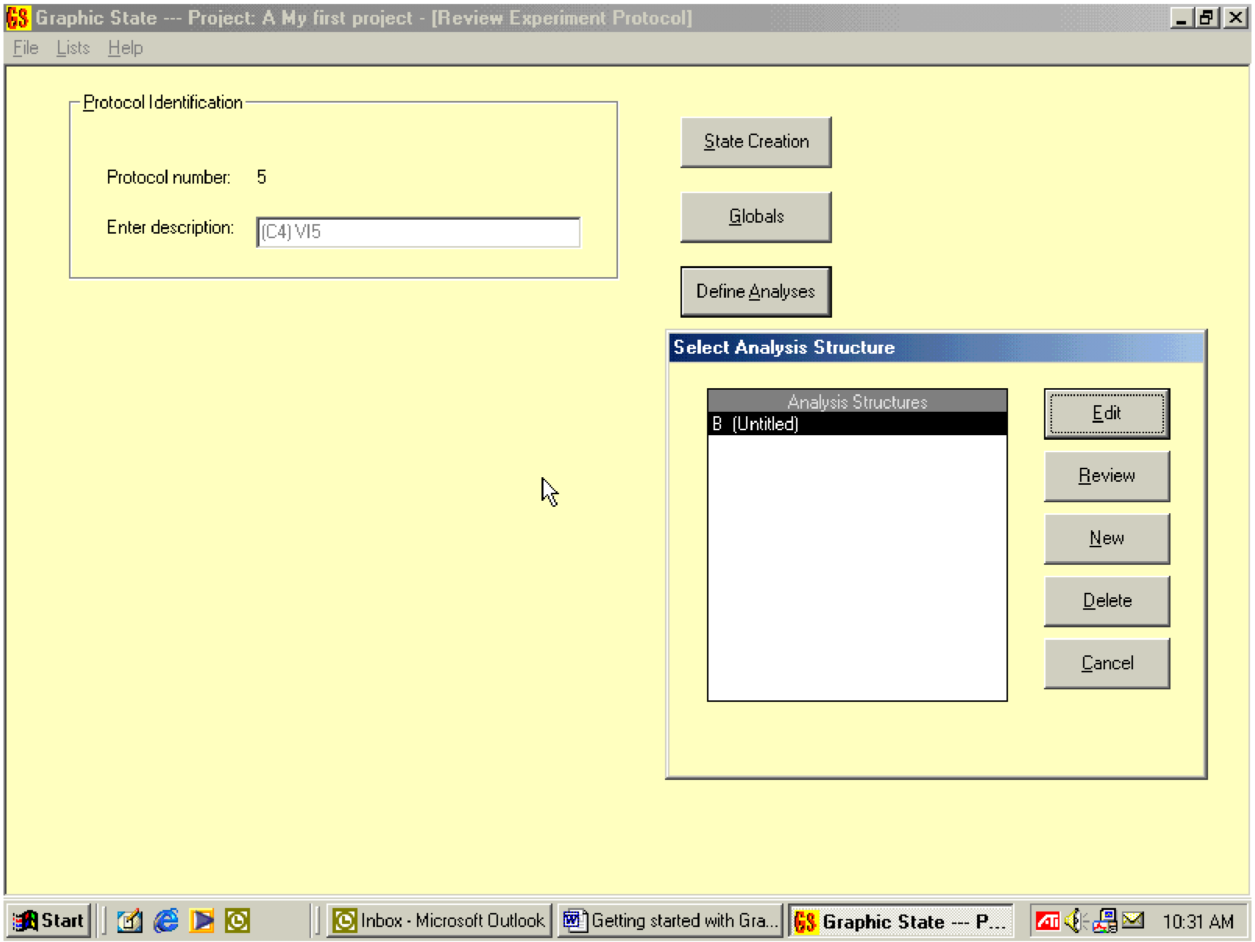
Proceed as usual on the run screen. Enter a subject ID and start the run. When the run finishes, immediately click on the SUBJ ID field again. The screen comes up for another subject ID. If you want to run a second subject with the same protocol, simply enter the new animal ID and click OK. In our case, however, we also want to change the protocol number. There is a drop-down list beside “Change protocol” with all the preloaded protocols. Select Protocol 5, make sure you have also entered an animal ID, and click OK.



When this second run ends, click on End/Terminate and analyze your data.

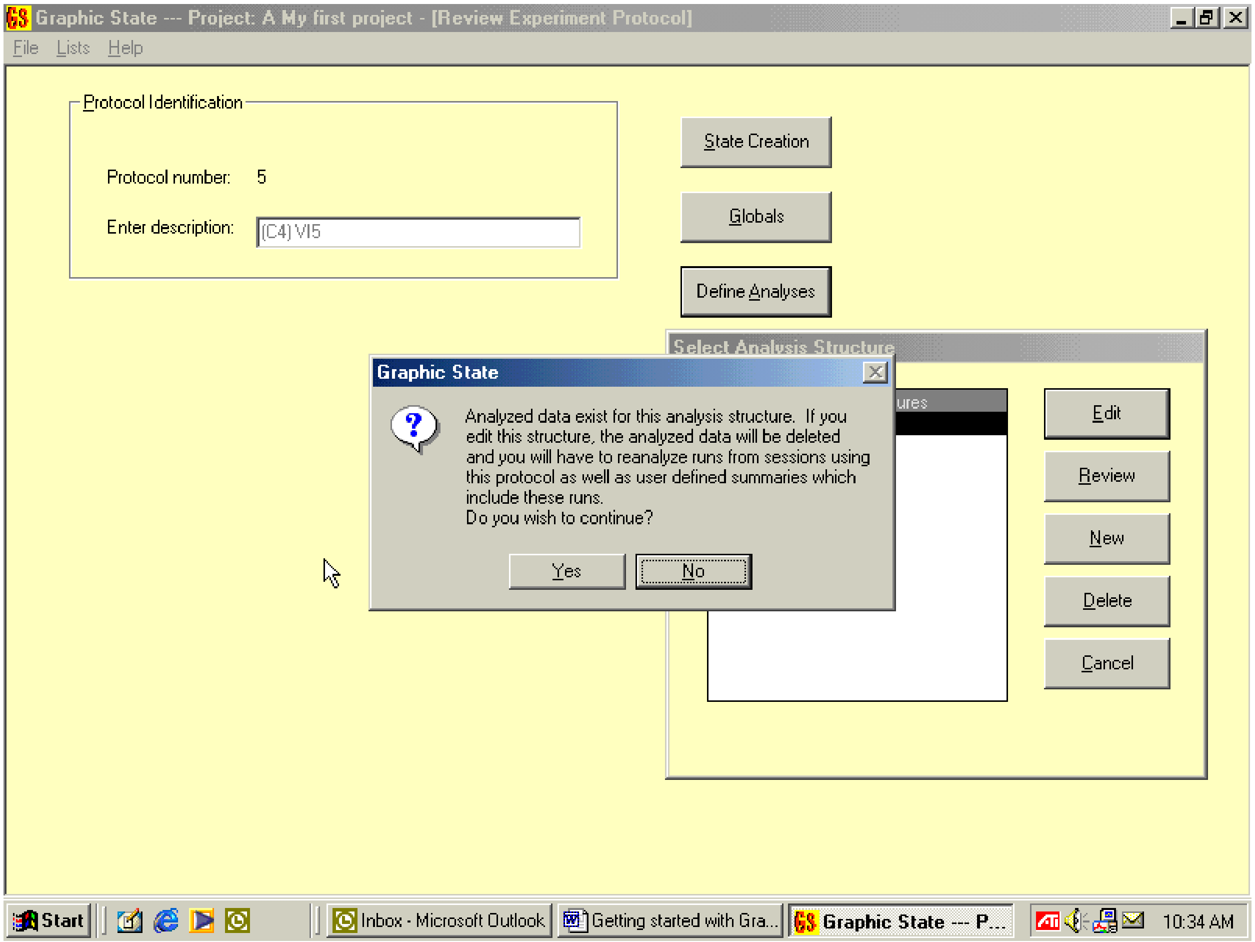
Now, let's add more data analysis elements to one of the protocols and reanalyze the data we've already collected. Select Edit/Review Experiment Protocol from the File menu. Notice that Protocols 2, 3, 4 and 5 are listed in black now. Select Protocol 5. The Edit button will be grayed out because you cannot change a protocol that has been run with non-pilot data. Click on the Review button and then on the Define Analyses button.

If you previously created a cumulative record analysis element here, you will now see a list of existing analysis structures (containing the default one named "B (Untitled)" unless you gave it a different name.

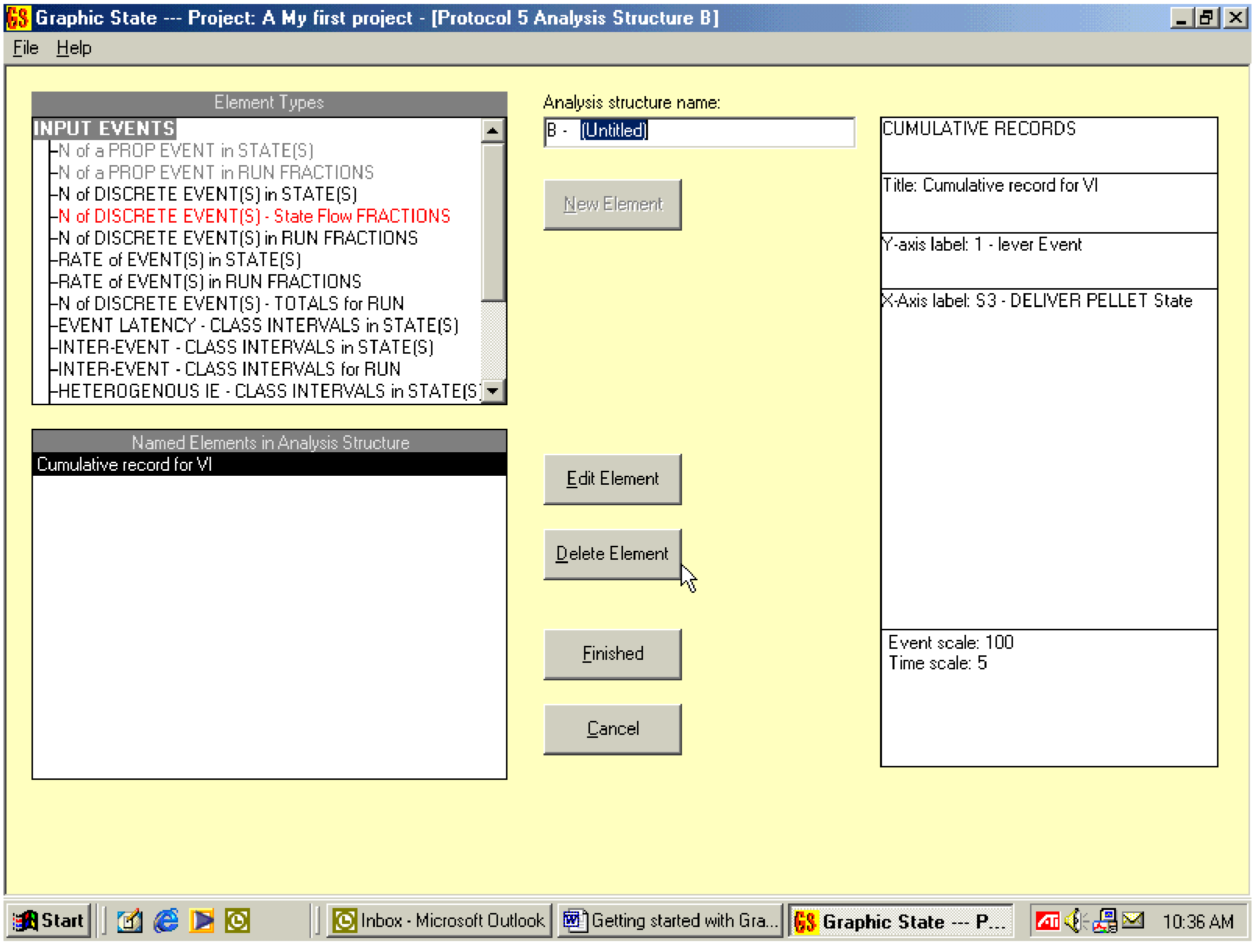


We are going to add more analysis elements, so you may *either* click on the New button to create a new analysis structure containing the new elements or you may click on the Edit button to change structure B. Click on the Edit button in this case.

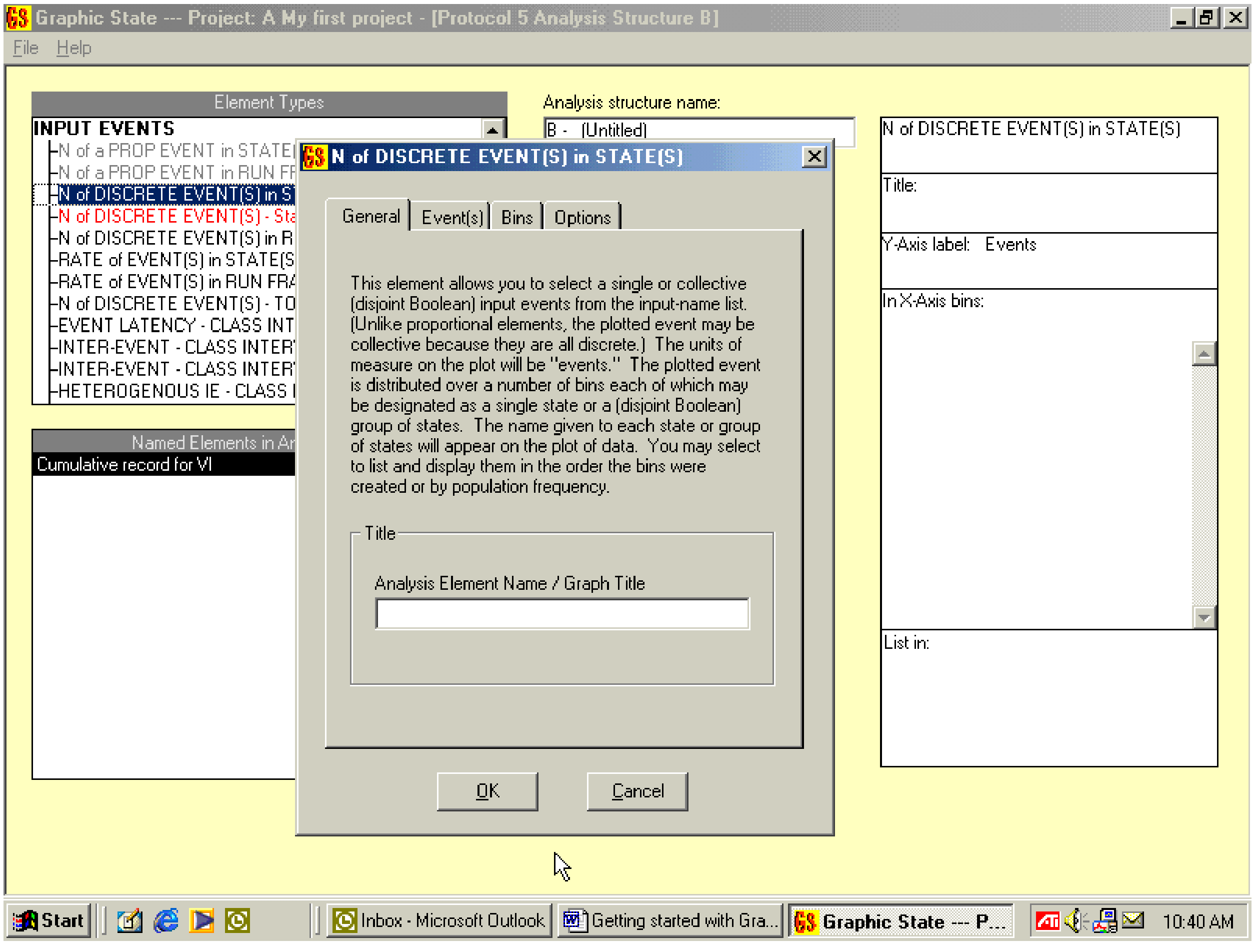
Because we have previously analyzed data using this structure, a warning message appears saying that, if you change the analysis structure, you will need to analyze the data again. That is acceptable, so click on the “Yes” button.



Now we see again the list of possible analysis elements as well as a definition of the one we already have set up.

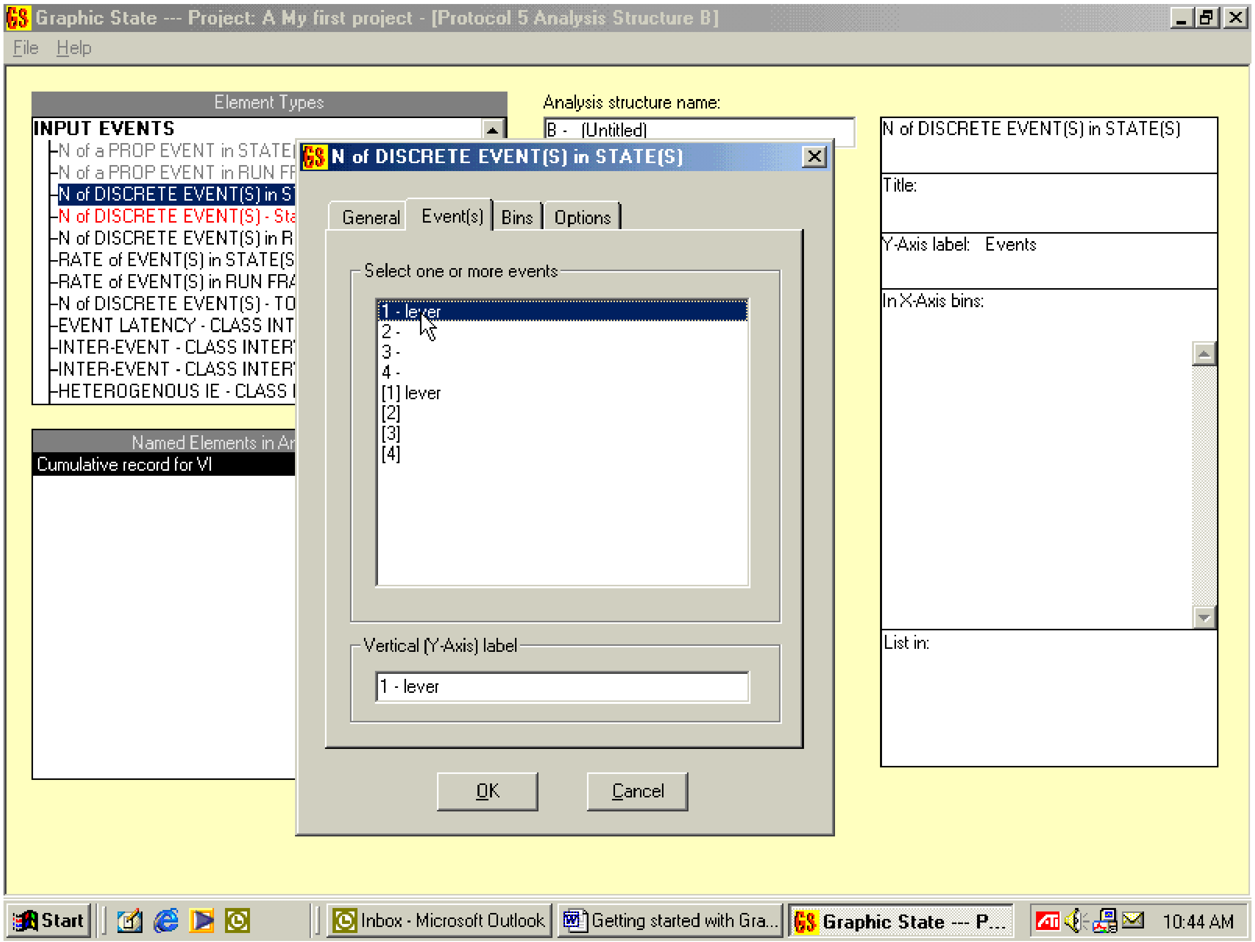


Let's investigate exactly where (i.e. in which state) the majority of the lever presses are occurring. Highlight the element type "N of DISCRETE EVENT(S) in STATE(S)" and click on the New Element button.

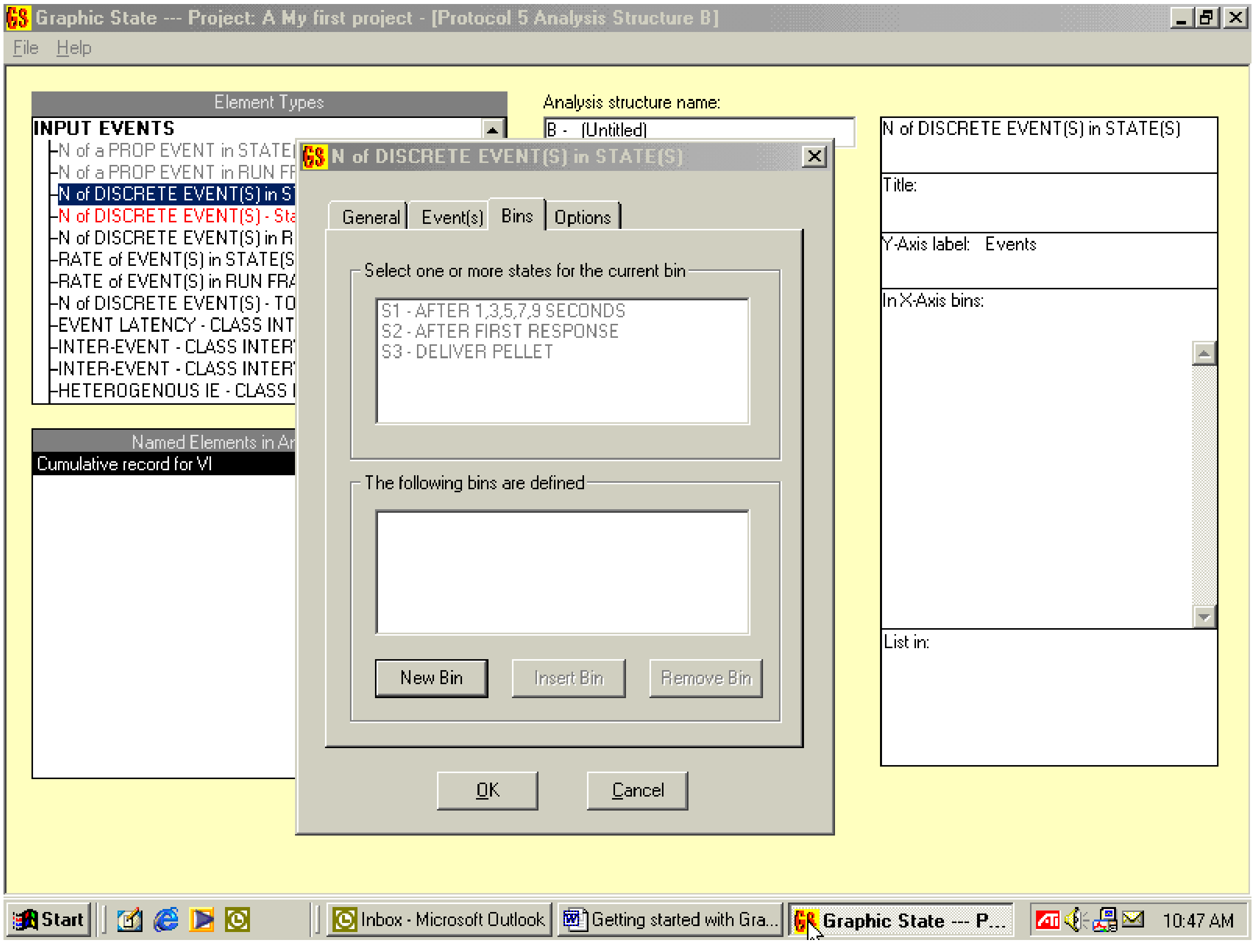


All of the element definition screens are similar in form, but they ask for the specific information required for the particular analysis. This one needs a name for the graph, the event or events to be counted, the state or states for each "bin" of the graph (i.e. each bar on the resulting histogram), and some options about the order in which to display the bins.

Enter a title on the "General" screen such as "Lever presses in each state." Click on the "Event(s) " tab. A list of eligible events (switch inputs), with the names you have assigned to them, will be presented. Select "1 – lever" and it will appear in the field called "Vertical (Y-Axis) label.

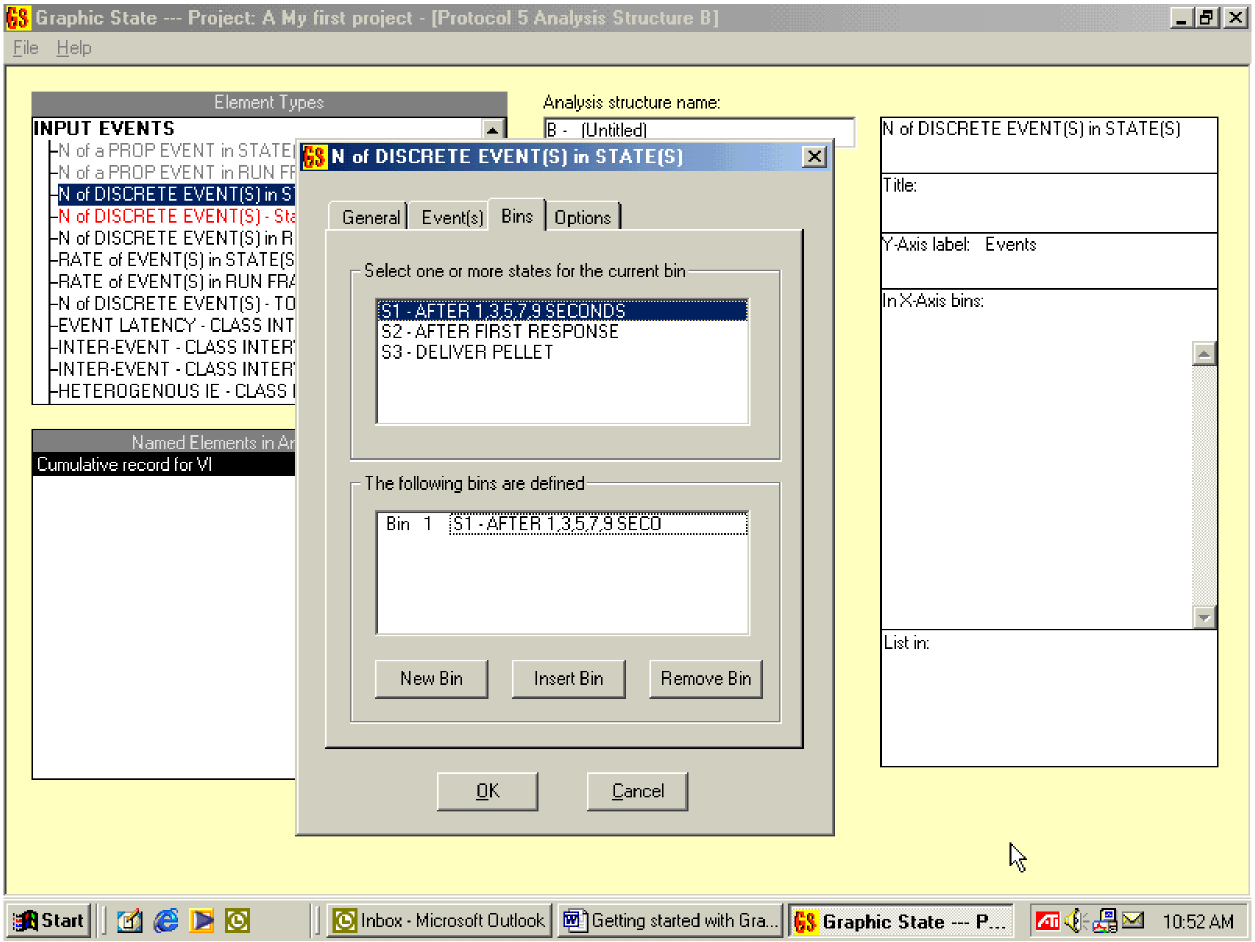


Next, click on the “Bins” tab. This will bring up a screen where you can define one or more categories/bars for the resulting graph. Initially, no bins are defined. The screen lists all of the states of your protocol, and you will select one or more states for each bin that you want to create.

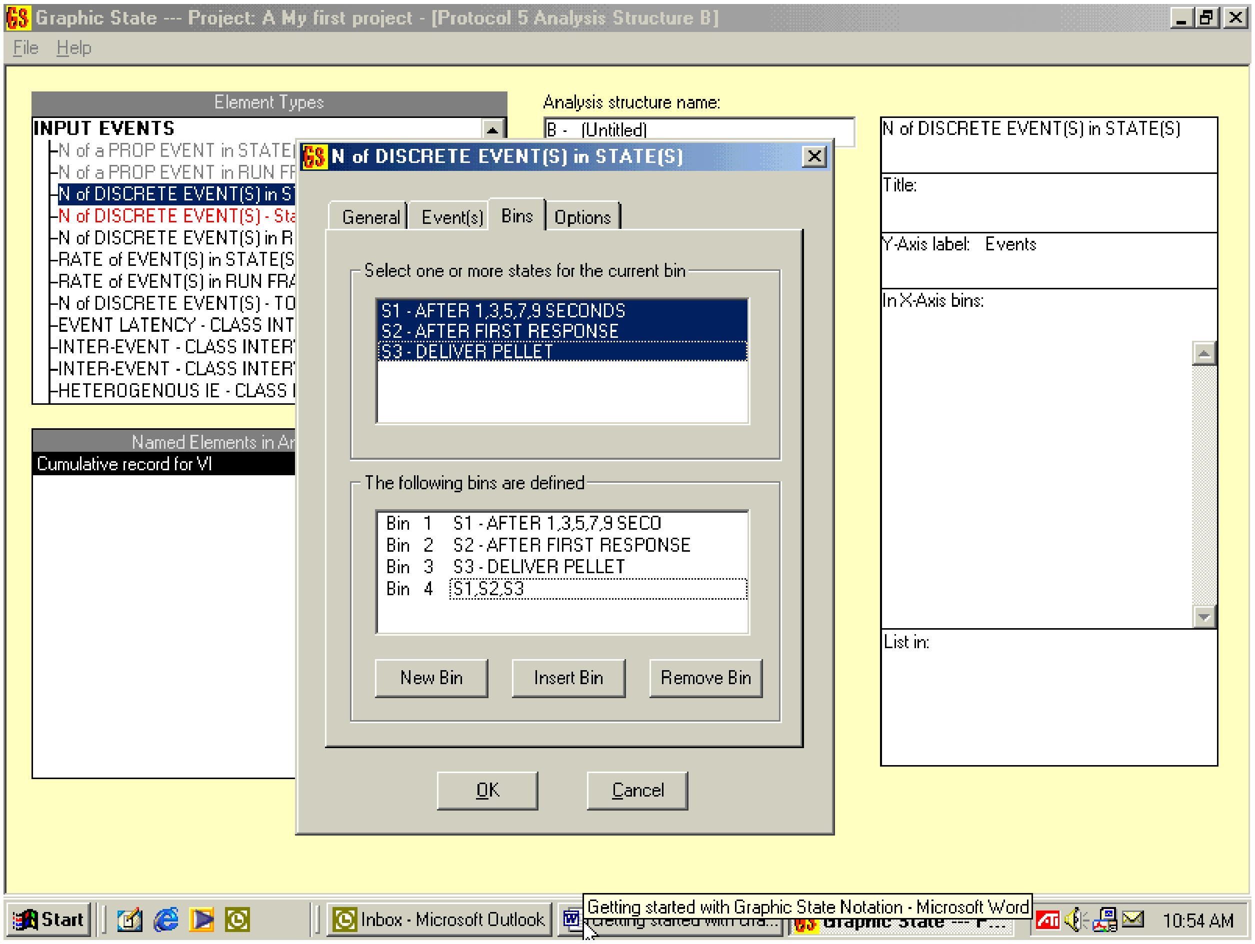


We will define four bins – one for each of the states S1, S2 and S3, and a fourth bin that includes all three states (and will report the total number of lever presses).

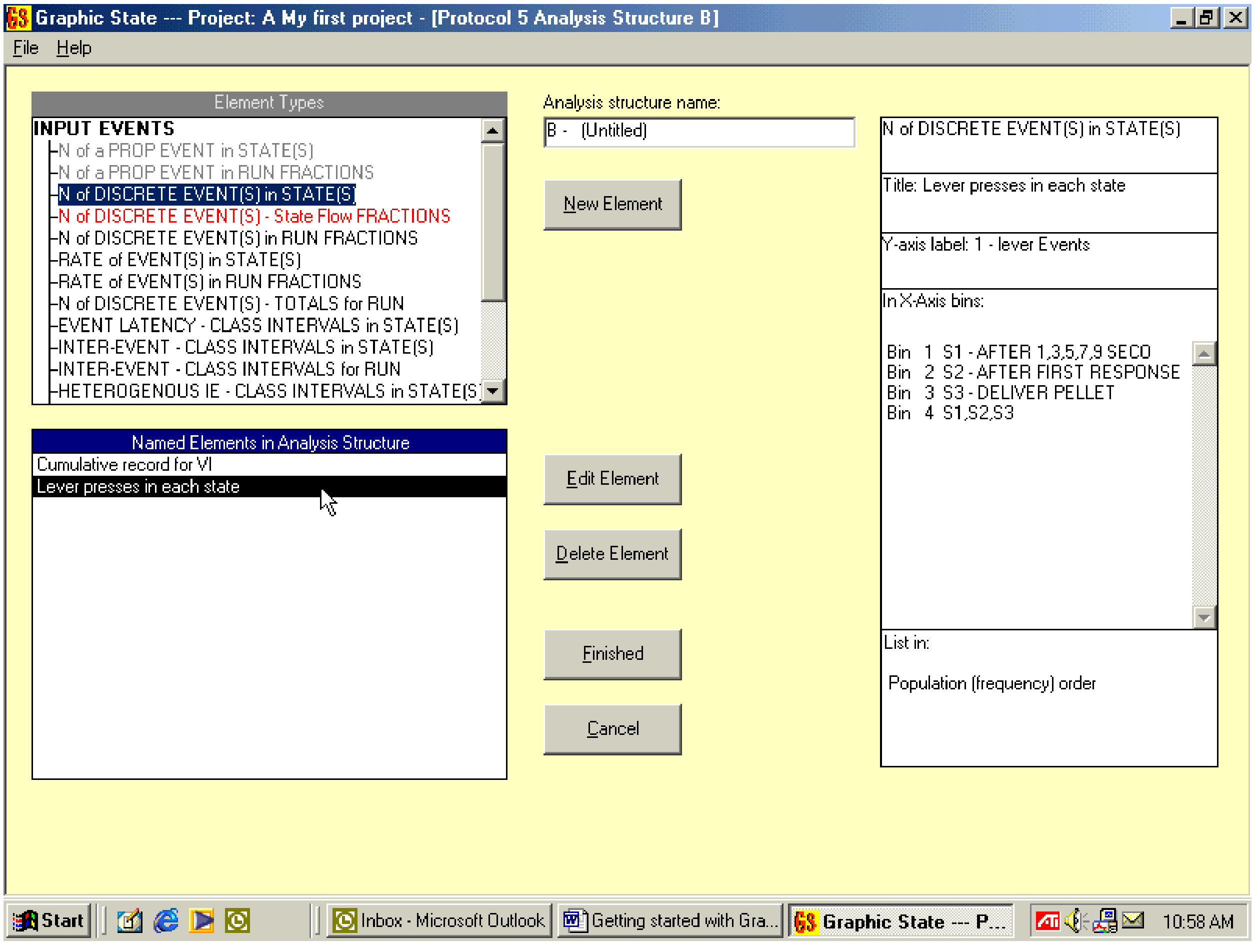
To define a bin, first click on the “New Bin” button and then select the first state.



Use the same procedure for states 2 and 3 (first click on "New Bin" and then select the state). Finally, create a fourth bin (click on "New Bin") and then select, in turn, all three states. Your screen should look like the one below.

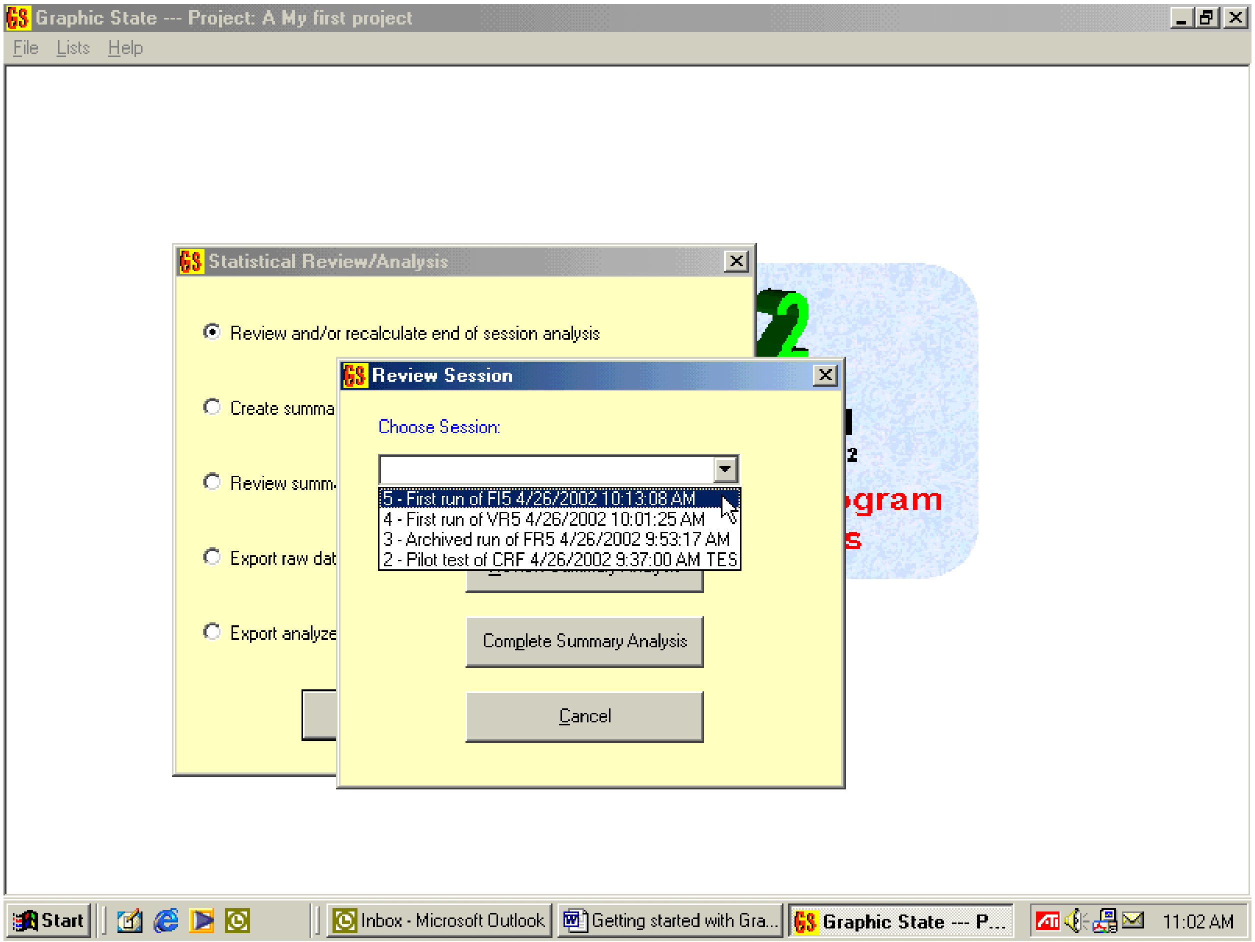


Click on the "Options" tab. You have two options for the order of the bins on the graph – either in the order in which we defined the bins, or in order by the frequency of the responses. Let's select "By population (frequency"). Finally, click on the OK button.



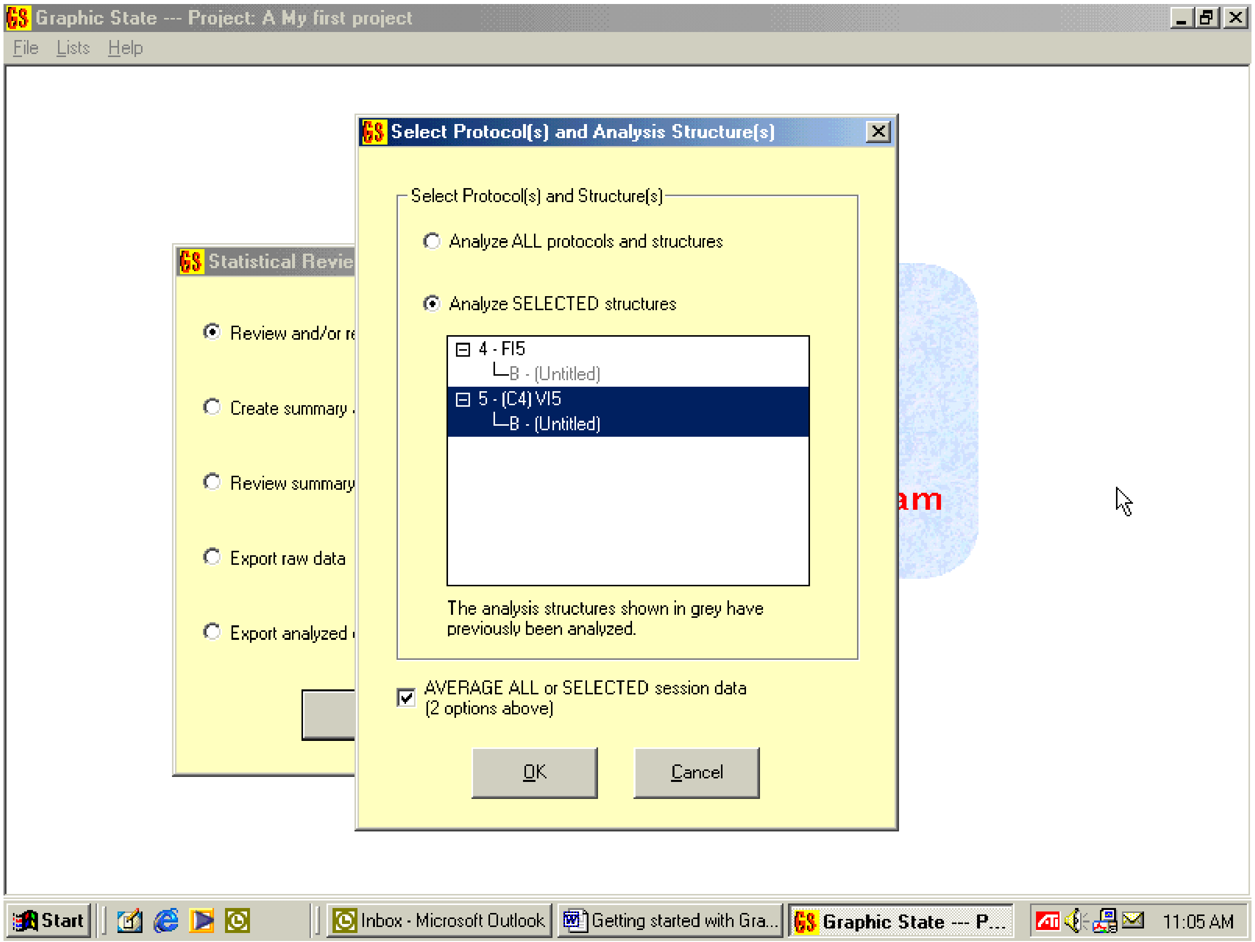
Now our new analysis element is listed in the lower left portion of the screen and a summary of it is listed on the right side of the screen (the highlighted element is always the one displayed on the right). Click on the Finished button and then on another Finished button to return to the main menu.

Select Statistical Review/Analysis from the File menu. Check the box labeled “Review and/or recalculate end of session analysis” and click OK. From the “Choose Session” drop-down list, select the session that included the run for Protocol 5. In my case, it was Session 5 that included a run of Protocol 4 and another of Protocol 5.



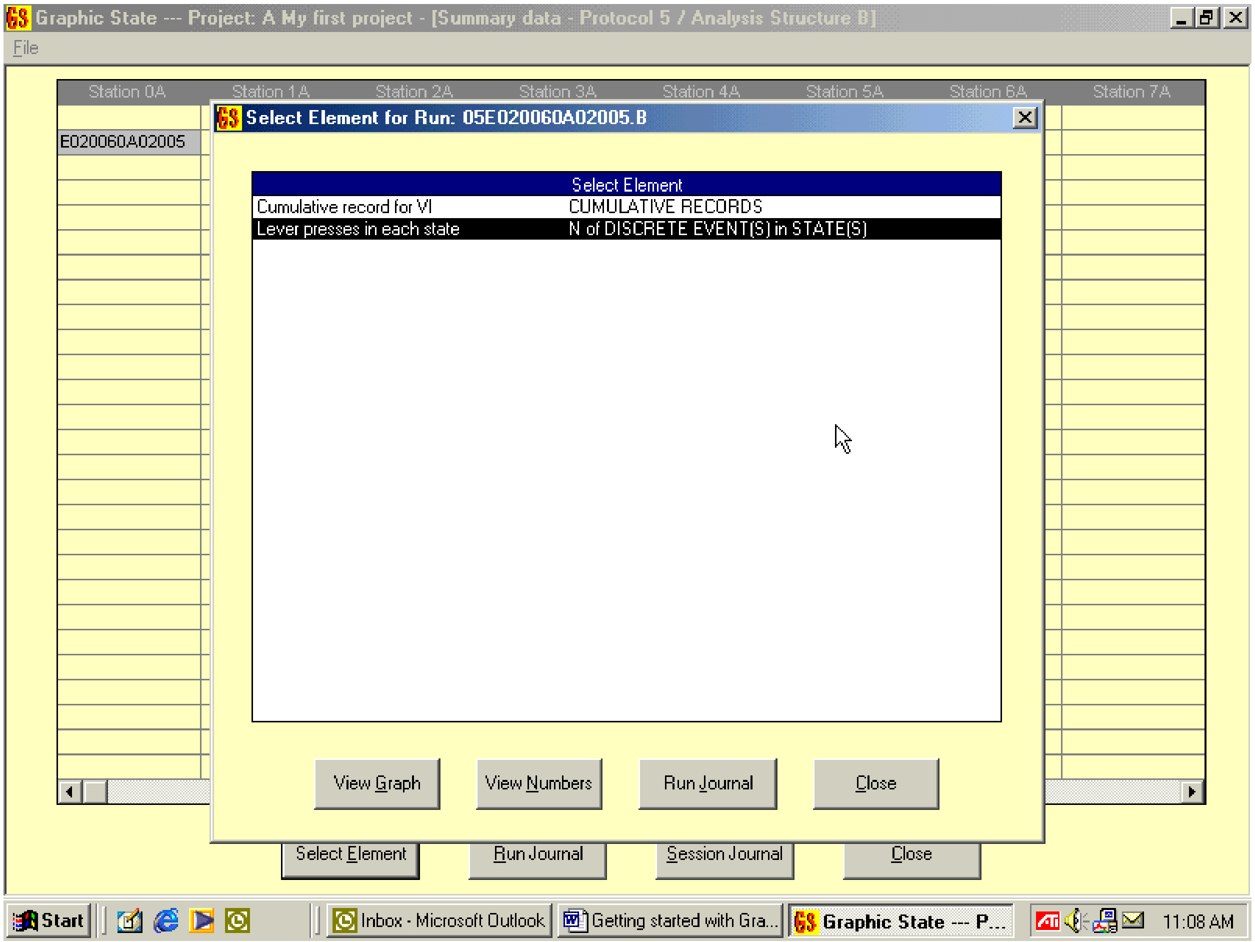
Click on the “Complete Summary Analysis” button.

We can select the option to “Analyze SELECTED structures” and then highlight Protocol 5, Structure B as shown in the following screen.

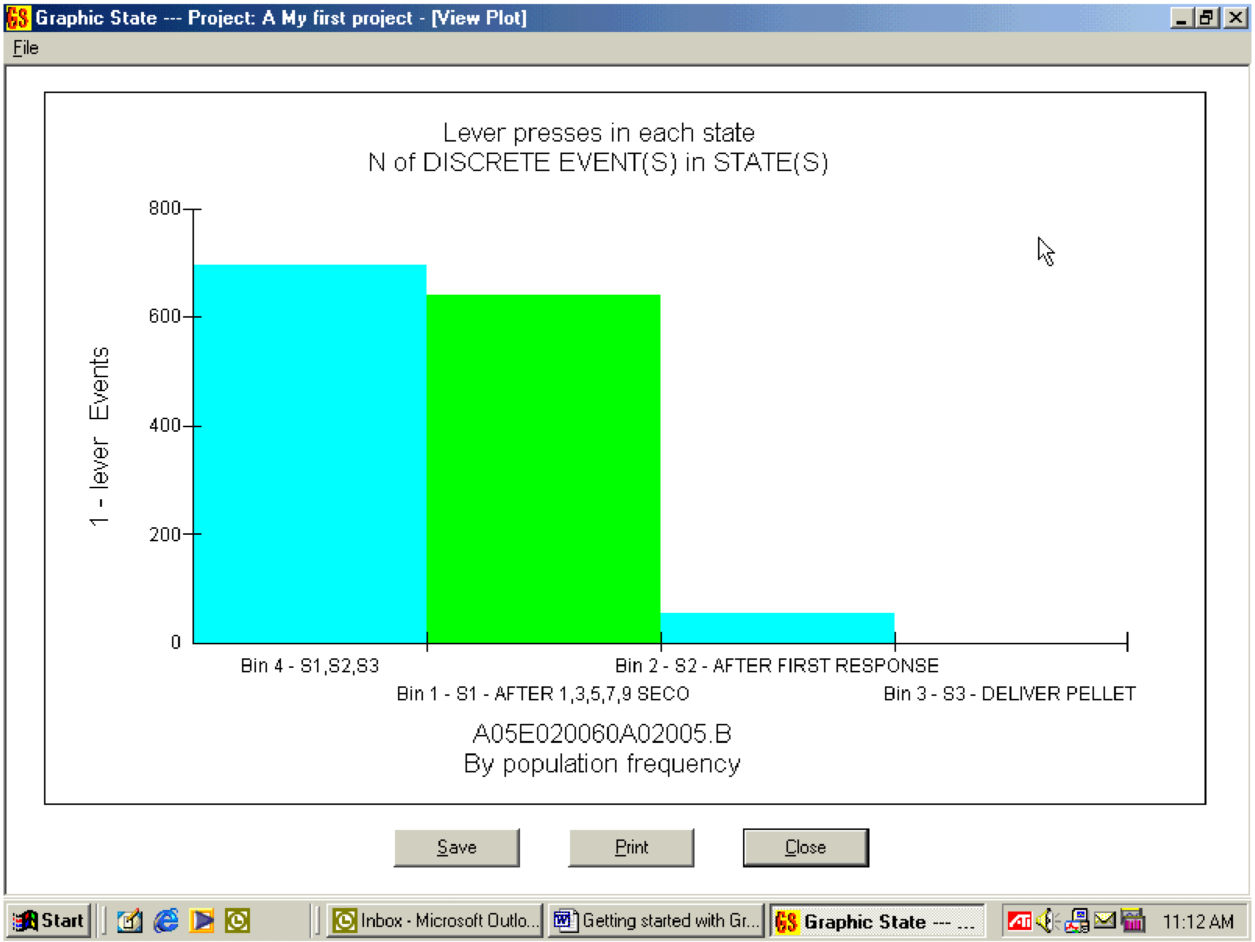


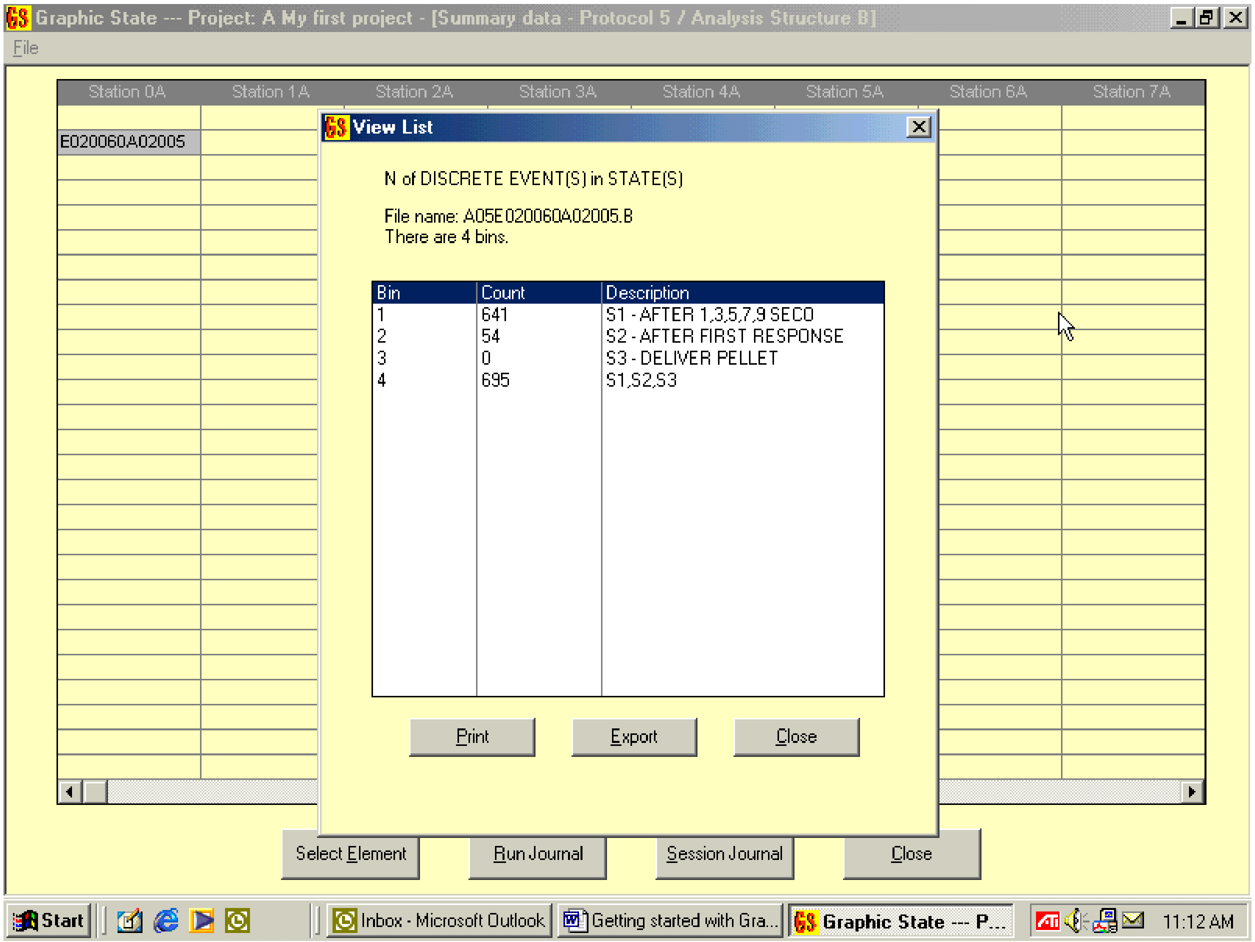
Click on OK and the analysis will proceed. Look at “Individual Runs” for Protocol 5 and select the only run that appears on the next screen. Notice that it appears in the second row under the header for Station 0A because it was the second run in that station for the session (the first run was using Protocol 4).

The next screen that appears lists both of the analysis elements that are available for us to view. Move the cursor to the second one – “Lever presses in each state” – and notice that now we have the option to “View Numbers” as well as to “View Graph.”



Both of these options are shown below.


Graphic State --- Project: A My first project - [View Plot]
File
Lever presses in each state
N of DISCRETE EVENT(S) in STATE(S)
800
600
400
200
0
1 - lever Events
Bin 4 - S1,S2,S3
Bin 1 - S1 - AFTER 1,3,5,7,9 SECO
Bin 2 - S2 - AFTER FIRST RESPONSE
Bin 3 - S3 - DELIVER PELLET
A05E020060A02005.B
By population frequency
Save
Print
Close
Start
Inbox - Microsoft Outlo...
Getting started with Gr...
Graphic State --- ...
11:12 AM

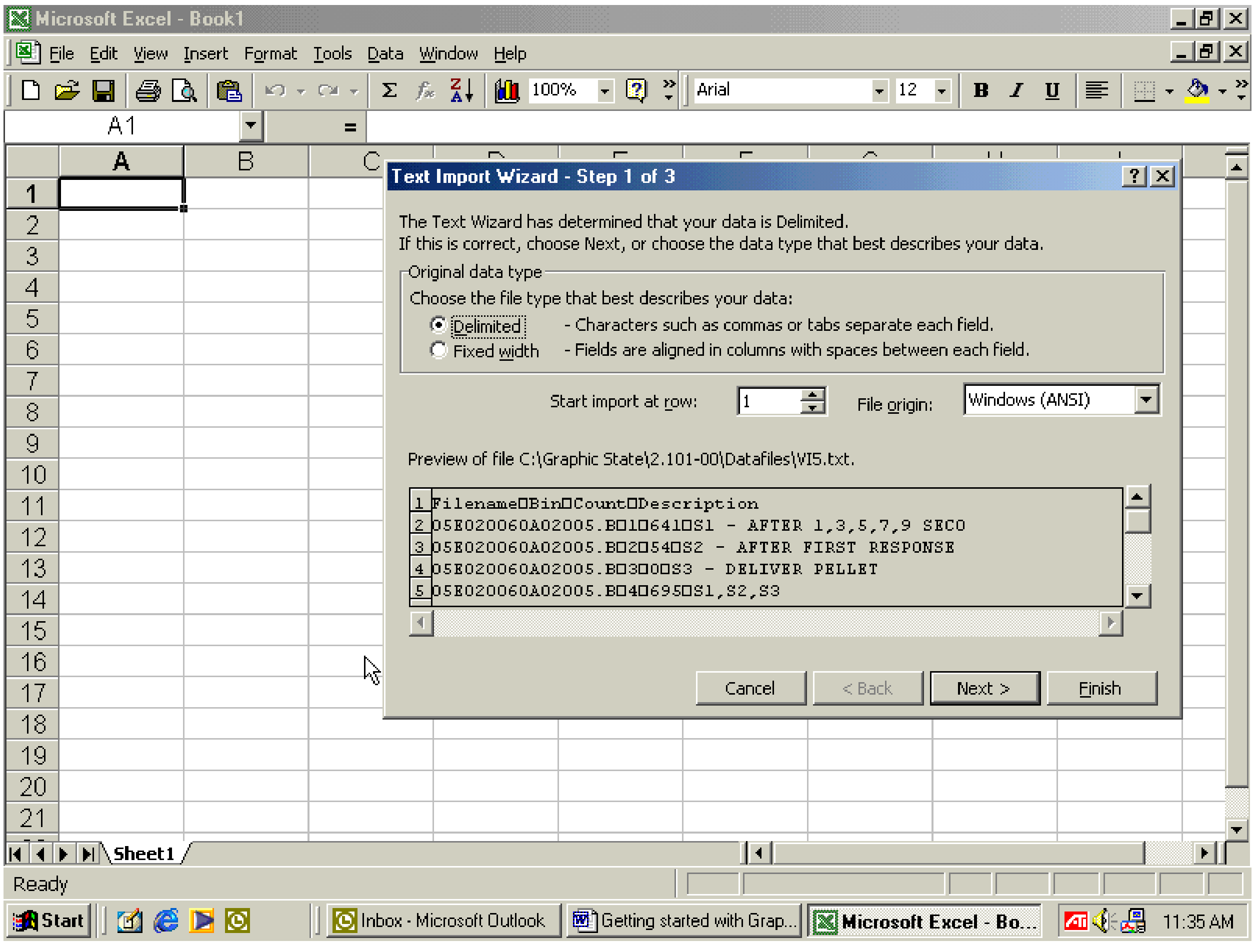


Exporting your data if necessary

You may want to export your data to process it further using a spreadsheet (e.g. Excel). There are several ways to export data. The first one is easy to demonstrate at this point.

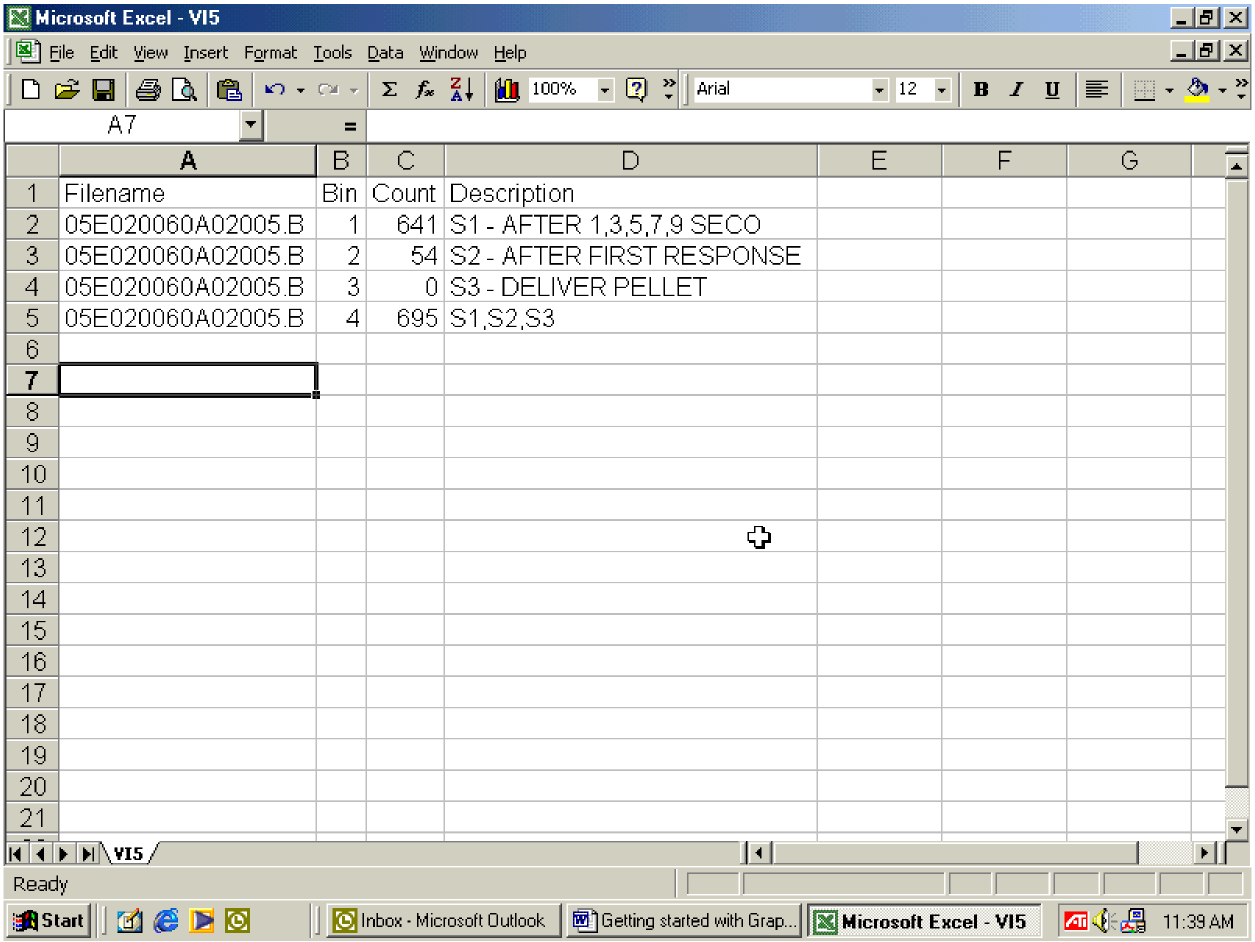
On the current screen, you have the option to Print the list of counts in each bin or to Export these counts. Click on the Export button. You will be prompted by Windows to supply a file name for the resulting file (a .txt file). Enter a name and click on the Save button. Now exit from the Graphic State program completely.

The best way to see what the exported data look like is to import the file into Excel (or a comparable spreadsheet program). Start Excel now and then open the text file you created in the last step. If you did not change the default directory, you will find the file in the Datafiles directory under your version number directory in the Graphic State directory. Remember that the file has an extension of “txt”, so you need to tell the Open dialogue in Excel to look for text files or for all files (not just for Excel files). The open command will invoke the Text Import Wizard, as shown below.



All of the default settings will work. Click on the Next button twice and then the Finish button.

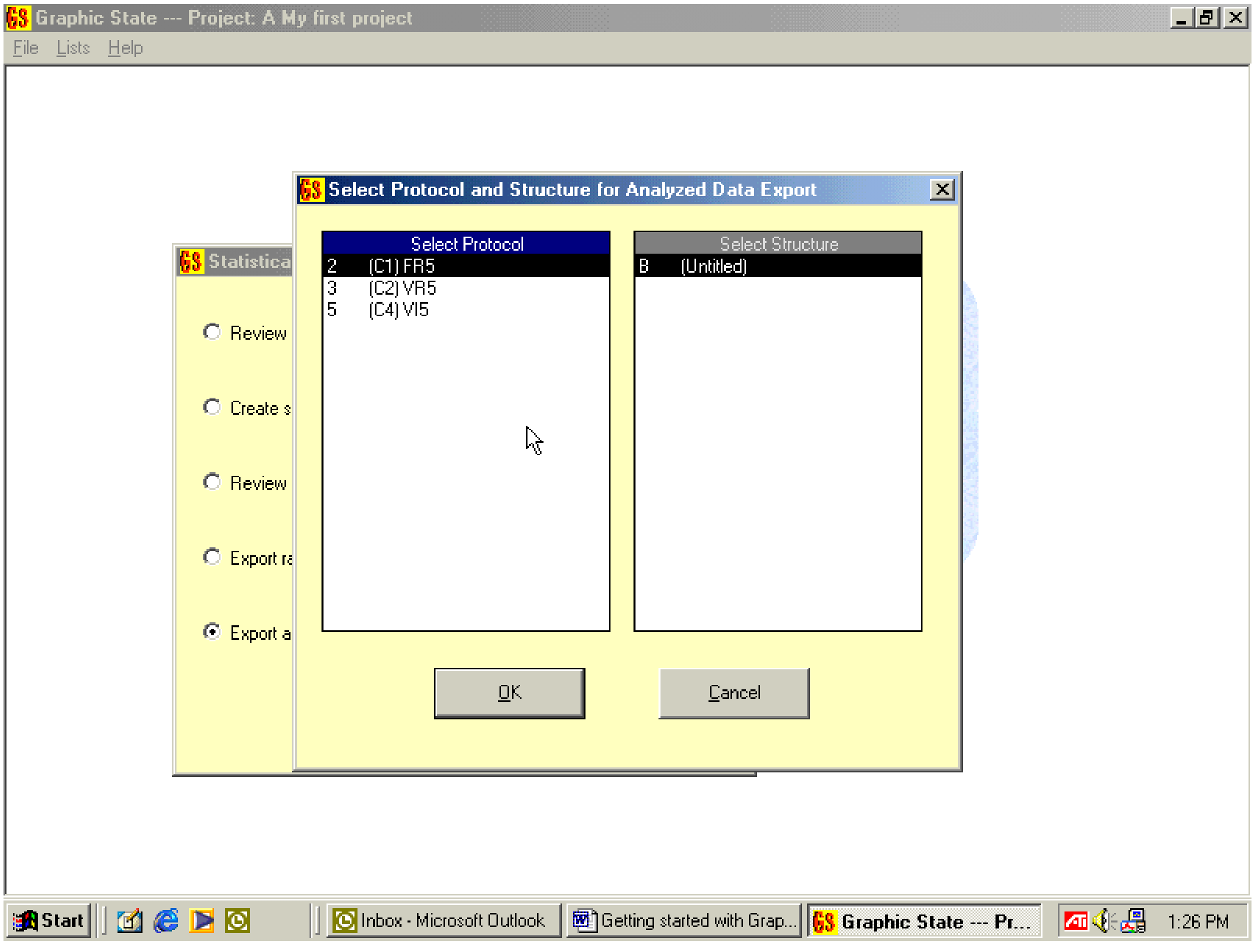
You may want to adjust the width of the columns to see the information better. You will see something like this.



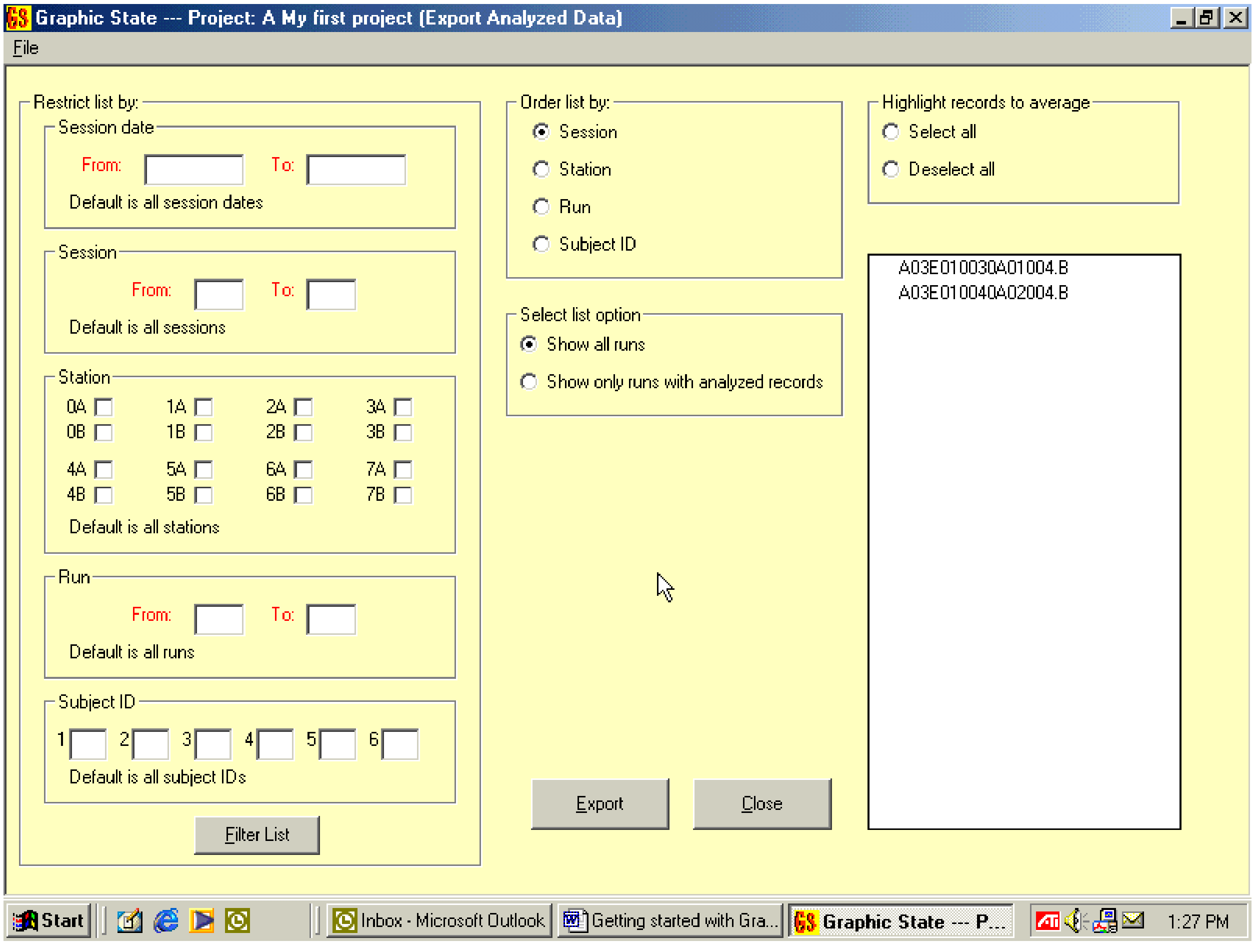
	A	B	C	D
1	Filename	Bin	Count	Description
2	05E020060A02005.B	1	641	S1 - AFTER 1,3,5,7,9 SECO
3	05E020060A02005.B	2	54	S2 - AFTER FIRST RESPONSE
4	05E020060A02005.B	3	0	S3 - DELIVER PELLET
5	05E020060A02005.B	4	695	S1,S2,S3

The data in this file can then be combined with those from other similar files and processed in any way you like.

There is another way to export data, this time from multiple runs. Select Statistical Review/Analysis from the File menu. Select "Export analyzed data (no Cum Records or Traces" and then click on the OK button. All protocols for which there are analyzed data (other than cumulative records or Multi-channel Event Traces, because these do not produce bins of counts) appear in a list.

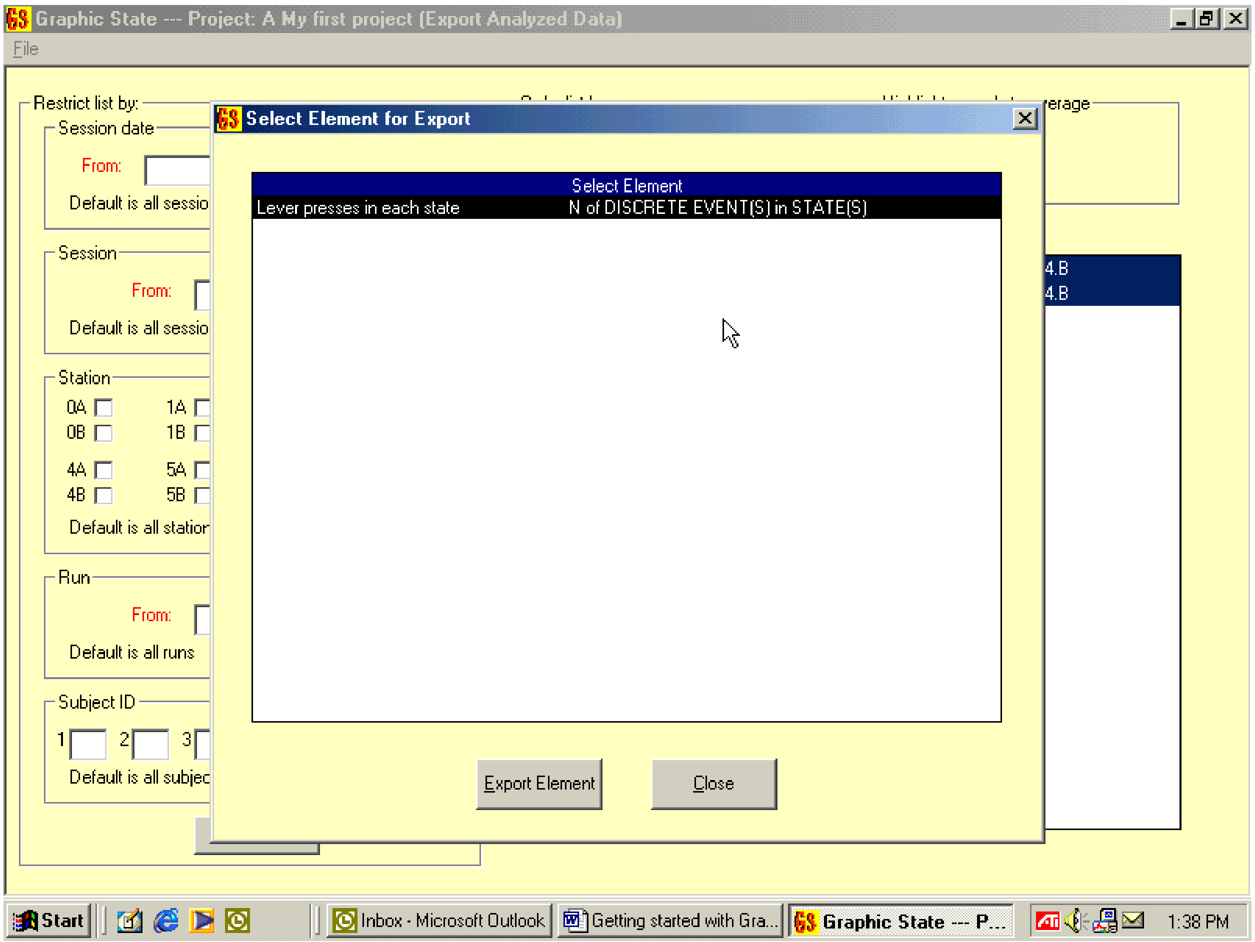


Select Protocol 3 and click on the OK button. A "filter" screen now appears which allows you to select all or only some of the runs for this protocol using a variety of criteria.



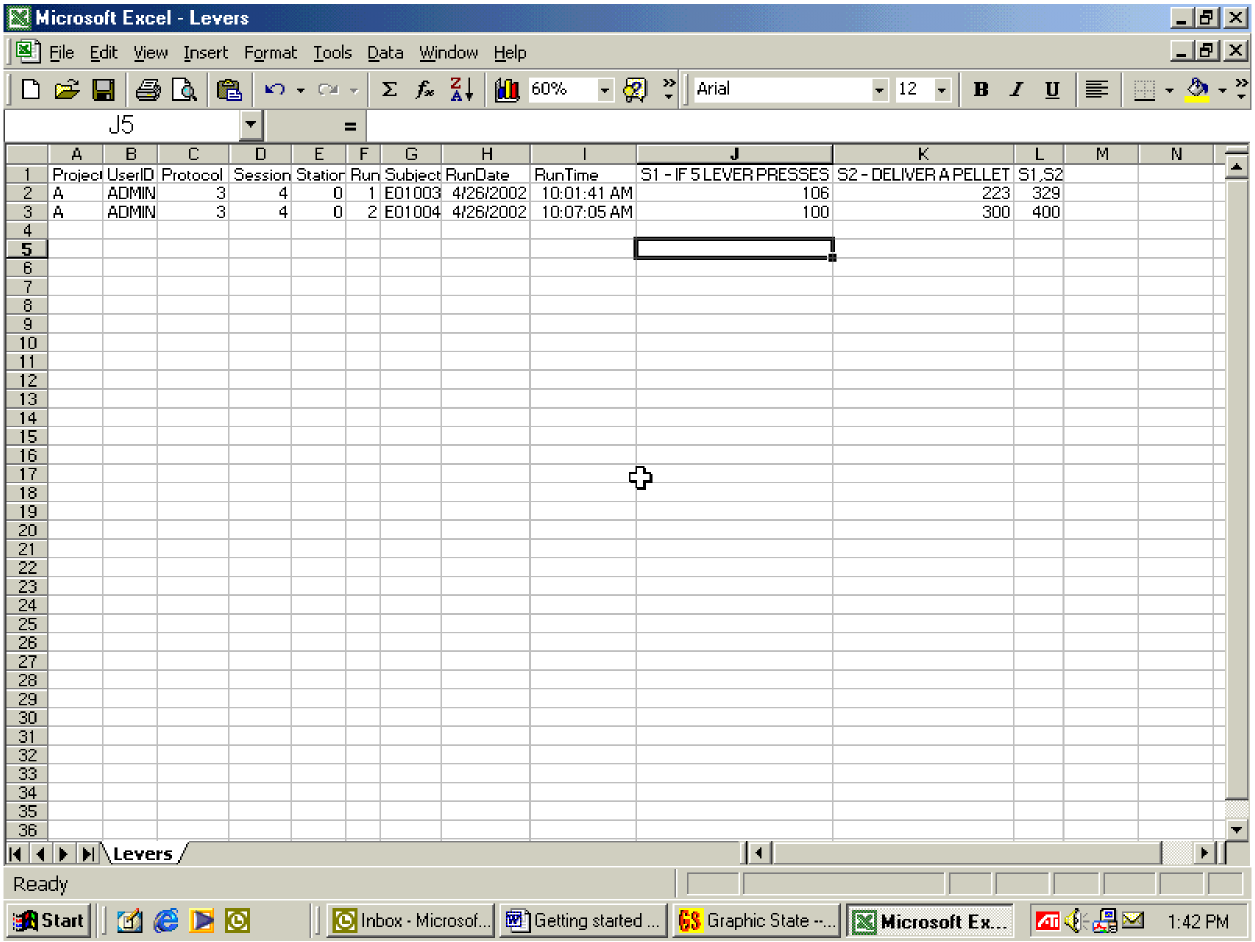
In the white area on the right side of the screen, you can see a list of all runs that qualify. In our example, there are only two, but imagine a much longer list containing runs for many different animals, in different stations, on different days.

By filling in selected portions of this screen, you can reduce the list of runs to only include those that meet your selection criteria. You may enter a range of session dates or of session numbers or of run numbers. You may select particular stations (by clicking in the appropriate check boxes). You may select runs where the subject ID matches the character or characters you type in positions 1 – 6 of the subject ID. For example, to select all animals from our sample experimental group 1 (i.e., all those with ID's that begin "E01"), fill in "E" in position 1, "0" in position 2 and "1" in position 3 and leave the last three positions blank. If you use any of the criteria in the left column on the screen, click on Filter List to remove runs that do not qualify from the list on the right side of the screen. You may select individual runs from the remaining list by clicking on the list items. You may also select all of them by clicking on the option "Select All" – which is what I will do. Click on the Export button, and you will see a list of analysis elements for which data may be exported.



Select the desired element (in this case there is only one) and then click on Export Element. Fill in the Windows prompt for a file name and click on Save.

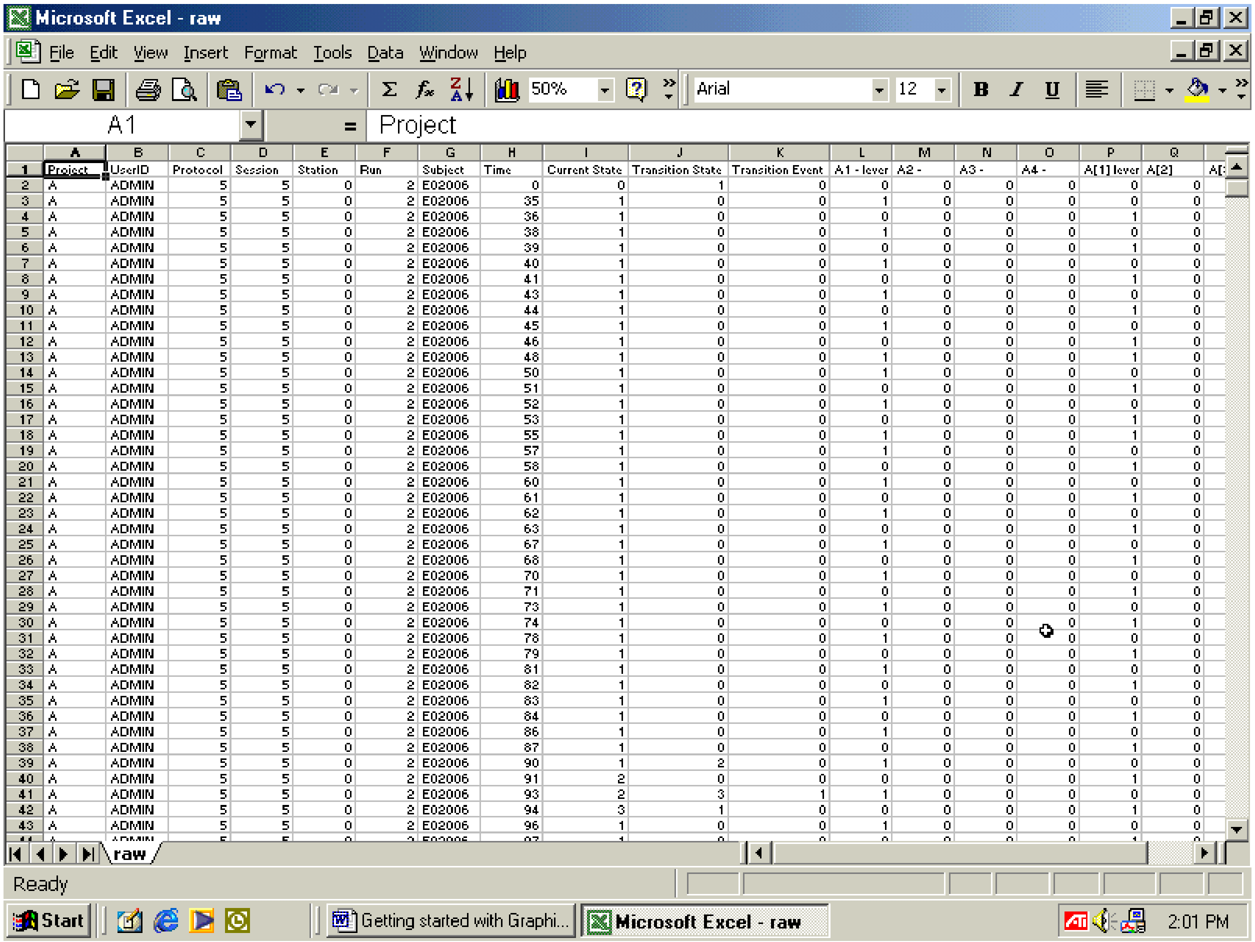
Open Excel again and import this new text file just as you did the last one. After adjusting the column widths, you will see something like this.



This will show you the data for the runs you selected with each animal as a separate row and each bin for the selected analysis element as a column. Other identifying information for each run is also included.

There may be an analysis you need to perform that is not provided as an analysis element built into Graphic State Notation. A common example of this is a trial-by-trial analysis of responses in a multi-trial protocol. In this kind of situation, you will want to export "raw" data and do the detailed processing yourself. The raw data available include a record (or a row in a spreadsheet) for each response onset or offset and for each state change during the run. Let's look at an example.

Select Statistical Review/Analysis from the File menu. Select the "Export raw data" option and click OK. Just as in the last example, select the appropriate protocol from the list that comes up (I selected Protocol 5, the VI5) and then use the filter screen to select specific runs. I selected the only run listed. The resulting text file, imported into Excel, looks like this (reduced in size to fit here).



	Project	UserID	Protocol	Session	Station	Run	Subject	Time	Current State	Transition State	Transition Event	A1 - lever	A2 -	A3 -	A4 -	A[1] lever	A[2]
2	A	ADMIN	5	5	0	2	E02006	0	0	1	0	0	0	0	0	0	0
3	A	ADMIN	5	5	0	2	E02006	35	1	0	0	1	0	0	0	0	0
4	A	ADMIN	5	5	0	2	E02006	36	1	0	0	0	0	0	0	1	0
5	A	ADMIN	5	5	0	2	E02006	38	1	0	0	1	0	0	0	0	0
6	A	ADMIN	5	5	0	2	E02006	39	1	0	0	0	0	0	0	1	0
7	A	ADMIN	5	5	0	2	E02006	40	1	0	0	1	0	0	0	0	0
8	A	ADMIN	5	5	0	2	E02006	41	1	0	0	0	0	0	0	1	0
9	A	ADMIN	5	5	0	2	E02006	43	1	0	0	1	0	0	0	0	0
10	A	ADMIN	5	5	0	2	E02006	44	1	0	0	0	0	0	0	1	0
11	A	ADMIN	5	5	0	2	E02006	45	1	0	0	1	0	0	0	0	0
12	A	ADMIN	5	5	0	2	E02006	46	1	0	0	0	0	0	0	1	0
13	A	ADMIN	5	5	0	2	E02006	48	1	0	0	1	0	0	0	1	0
14	A	ADMIN	5	5	0	2	E02006	50	1	0	0	1	0	0	0	0	0
15	A	ADMIN	5	5	0	2	E02006	51	1	0	0	0	0	0	0	1	0
16	A	ADMIN	5	5	0	2	E02006	52	1	0	0	1	0	0	0	0	0
17	A	ADMIN	5	5	0	2	E02006	53	1	0	0	0	0	0	0	1	0
18	A	ADMIN	5	5	0	2	E02006	55	1	0	0	1	0	0	0	1	0
19	A	ADMIN	5	5	0	2	E02006	57	1	0	0	1	0	0	0	0	0
20	A	ADMIN	5	5	0	2	E02006	58	1	0	0	0	0	0	0	1	0
21	A	ADMIN	5	5	0	2	E02006	60	1	0	0	1	0	0	0	0	0
22	A	ADMIN	5	5	0	2	E02006	61	1	0	0	0	0	0	0	1	0
23	A	ADMIN	5	5	0	2	E02006	62	1	0	0	1	0	0	0	0	0
24	A	ADMIN	5	5	0	2	E02006	63	1	0	0	0	0	0	0	1	0
25	A	ADMIN	5	5	0	2	E02006	67	1	0	0	1	0	0	0	0	0
26	A	ADMIN	5	5	0	2	E02006	68	1	0	0	0	0	0	0	1	0
27	A	ADMIN	5	5	0	2	E02006	70	1	0	0	1	0	0	0	0	0
28	A	ADMIN	5	5	0	2	E02006	71	1	0	0	0	0	0	0	1	0
29	A	ADMIN	5	5	0	2	E02006	73	1	0	0	1	0	0	0	0	0
30	A	ADMIN	5	5	0	2	E02006	74	1	0	0	0	0	0	0	1	0
31	A	ADMIN	5	5	0	2	E02006	78	1	0	0	1	0	0	0	0	0
32	A	ADMIN	5	5	0	2	E02006	79	1	0	0	0	0	0	0	1	0
33	A	ADMIN	5	5	0	2	E02006	81	1	0	0	1	0	0	0	0	0
34	A	ADMIN	5	5	0	2	E02006	82	1	0	0	0	0	0	0	1	0
35	A	ADMIN	5	5	0	2	E02006	83	1	0	0	1	0	0	0	0	0
36	A	ADMIN	5	5	0	2	E02006	84	1	0	0	0	0	0	0	1	0
37	A	ADMIN	5	5	0	2	E02006	86	1	0	0	1	0	0	0	0	0
38	A	ADMIN	5	5	0	2	E02006	87	1	0	0	0	0	0	0	1	0
39	A	ADMIN	5	5	0	2	E02006	90	1	2	0	1	0	0	0	0	0
40	A	ADMIN	5	5	0	2	E02006	91	2	0	0	0	0	0	0	1	0
41	A	ADMIN	5	5	0	2	E02006	93	2	3	1	1	0	0	0	0	0
42	A	ADMIN	5	5	0	2	E02006	94	3	1	0	0	0	0	0	1	0
43	A	ADMIN	5	5	0	2	E02006	96	1	0	0	1	0	0	0	0	0

The first seven columns do not change during a single run; they contain identifying information about the project, user, protocol, session, station, run and animal ID.

The next column, labeled "Time", is the time stamp (in sample units since the beginning of the run) for each event or transition recorded in this file. In our case, the sample unit is 100 ms or 0.1 seconds. Therefore, to convert the times to seconds since the start of the run, divide each time by 10.

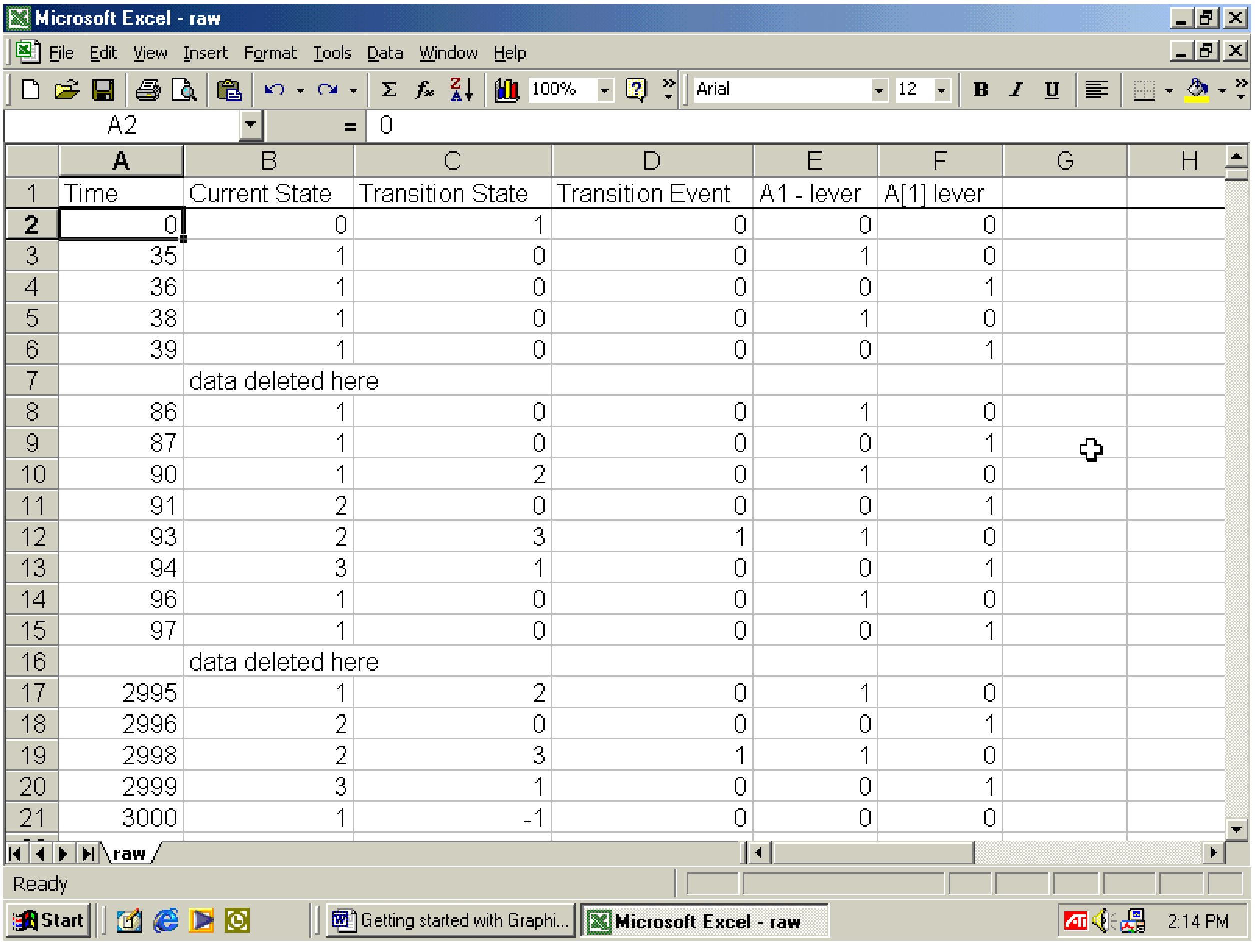
The "Current State" column lists the state number that was active at each time. State 0 is the RDY (Ready) state and State –1 is the FIN state.

The "Transition State" column shows the state to which control was shifted if the current line represents a state transition; it is zero if no transition occurred at this time.

The “Transition Event” column contains a “1” if a state transition occurred because of an input event (response); it is “0” if no transition occurred or if a transition occurred because of time or the number of entries.

The next four (in this case) columns record the presence (“1”) or absence (“0”) of a response onset on each of the possible input switches. The following four columns record the presence or absence of a response *offset.* Response onsets are noted with column headings like “A1 – lever” and offsets are noted with column headings like “A[1] lever”. For Linc configurations where there are more than four switch inputs, there will be more columns in the raw data.

Let’s look in more detail at the most relevant columns in this spreadsheet. I have deleted the leading identification columns and all of the switch columns except the A1 onset and offset columns (so that what remains will fit nicely on the screen and we can read it here easily). I also deleted *many* rows; raw data files get very big.



	A	B	C	D	E	F
1	Time	Current State	Transition State	Transition Event	A1 - lever	A[1] lever
2	0	0	1	0	0	0
3	35	1	0	0	1	0
4	36	1	0	0	0	1
5	38	1	0	0	1	0
6	39	1	0	0	0	1
7		data deleted here				
8	86	1	0	0	1	0
9	87	1	0	0	0	1
10	90	1	2	0	1	0
11	91	2	0	0	0	1
12	93	2	3	1	1	0
13	94	3	1	0	0	1
14	96	1	0	0	1	0
15	97	1	0	0	0	1
16		data deleted here				
17	2995	1	2	0	1	0
18	2996	2	0	0	0	1
19	2998	2	3	1	1	0
20	2999	3	1	0	0	1
21	3000	1	-1	0	0	0

In the first row, we can see that the run began at offset time 0 (by definition) in State 0 (RDY) and there was a transition to State 1. All runs begin this way.

Next we see pairs of entries in which the lever was pressed (e.g., at 3.5 seconds into the run) and then released (at 3.6 seconds into the run). This continues for a period of time (a time selected from the list for this VI); I have deleted these entries for brevity. At 9 seconds into the run (row 10 above), there is a transition to State 2. From this information, we can compute the value of the first variable interval to be 9 seconds (or 90 sample units).

There was also a lever press recorded at time 90. However, this lever press did not cause the transition to State 2 (it was a time transition), and since it occurred in State 1, it also did not cause the delivery of the reinforcer. That happened at time 93 (row 12 above) where we see that a transition to State 3 was caused by an event (the

“1” in the “Transition Event” column) and we see that a lever press response was recorded. At the next time marker (94), a transition back to State 1 occurred. This is the start of the next “trial”.

The last line in the spreadsheet, at time marker 3000 (3000 sample units = 300 seconds = 5 minutes), records a final transition to State –1 (FIN).

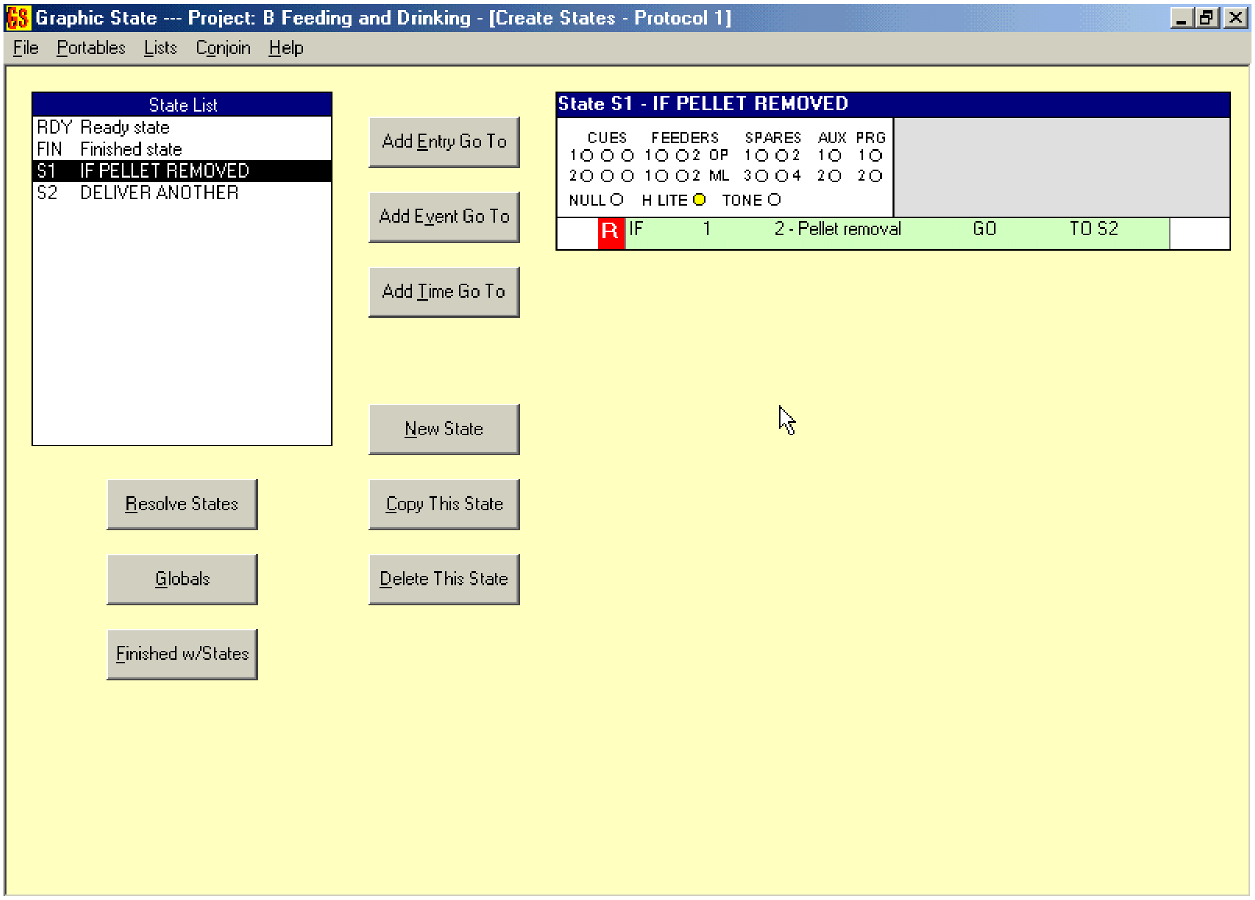
For anything other than trivial processing, you will probably want to create Excel macros that you can use repeatedly. A sample macro is included in the next section of this manual in the example called “Multiple trials of different types.”

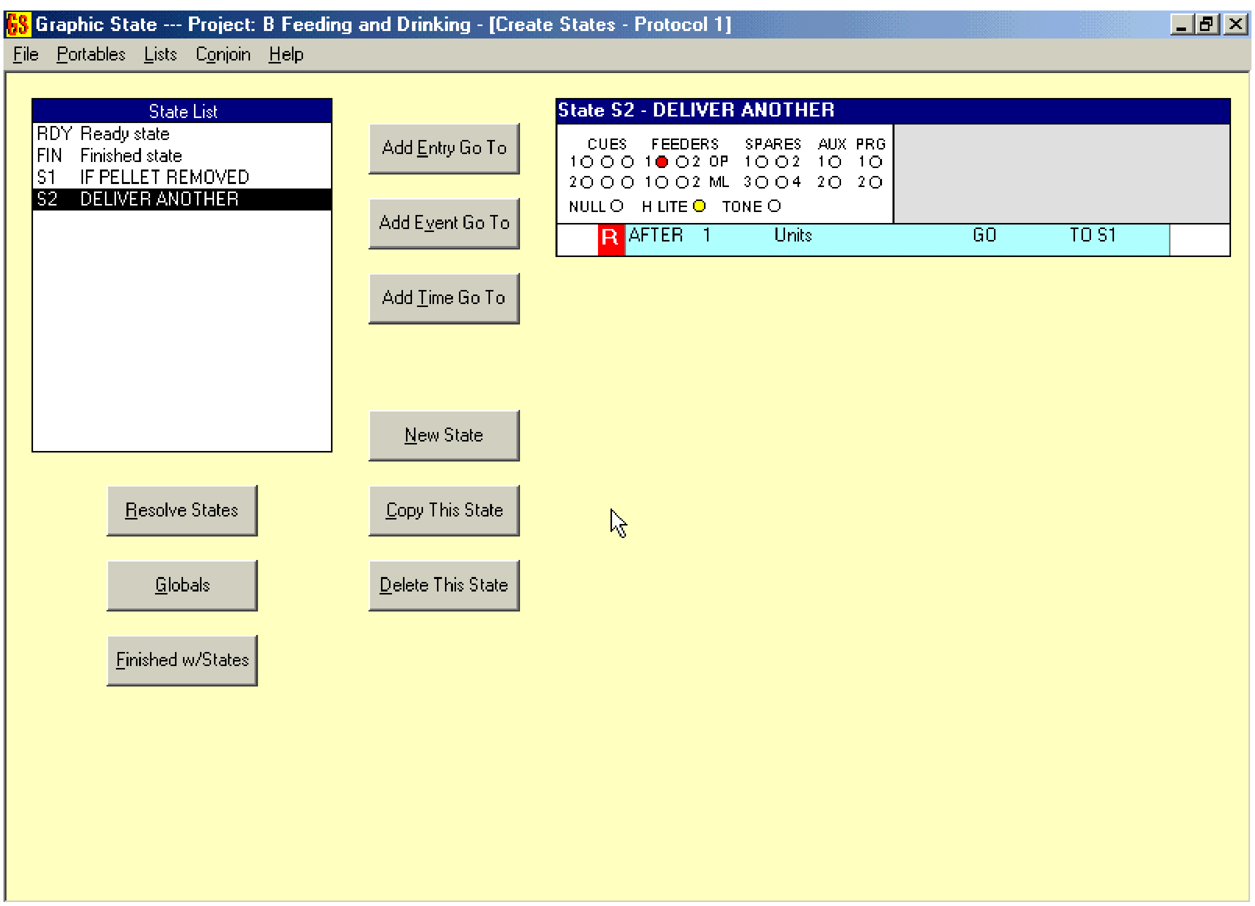
Some additional examples

Meal pattern analysis

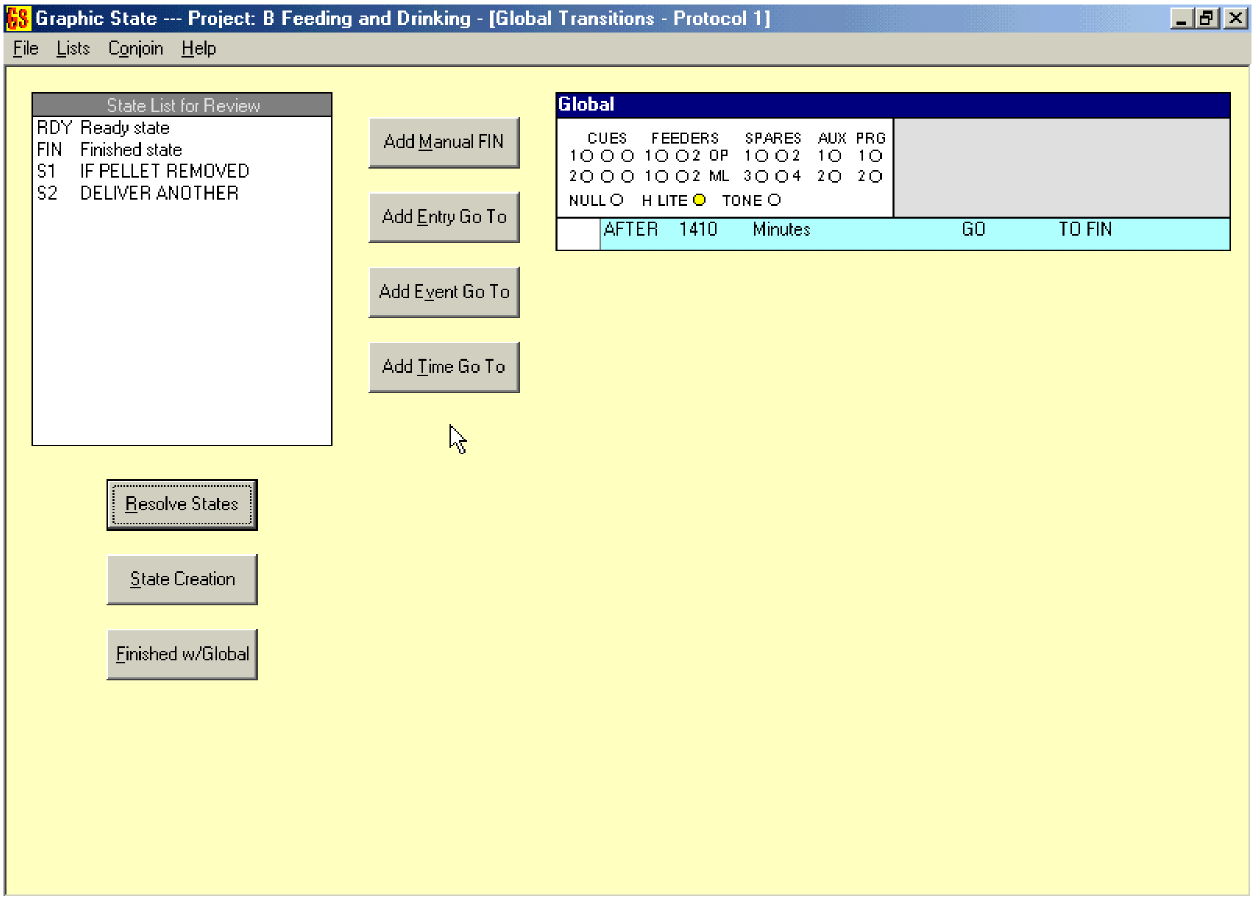
A feeding and drinking application usually has a very simple state flow with all the complexity in the analysis. What I will show here is a free feeding and drinking protocol where water is available at all times (with licks detected by a photocell mounted on a Coulbourn lickometer) and a new food pellet is delivered each time one is removed from the pellet trough (also detected by a photocell mounted on the pellet trough). I'll then demonstrate some of the data analysis possibilities.

If you are using our "standard" pellet trough, the protocol only needs two states – one where pellet removals are detected and one where new pellets are delivered. If you use our auto-feeding trough, you only need one state (to control the total length of time of the run) because the trough will automatically trigger the dispensing of a new pellet when one is removed. Assuming the standard trough, the two states look like this.

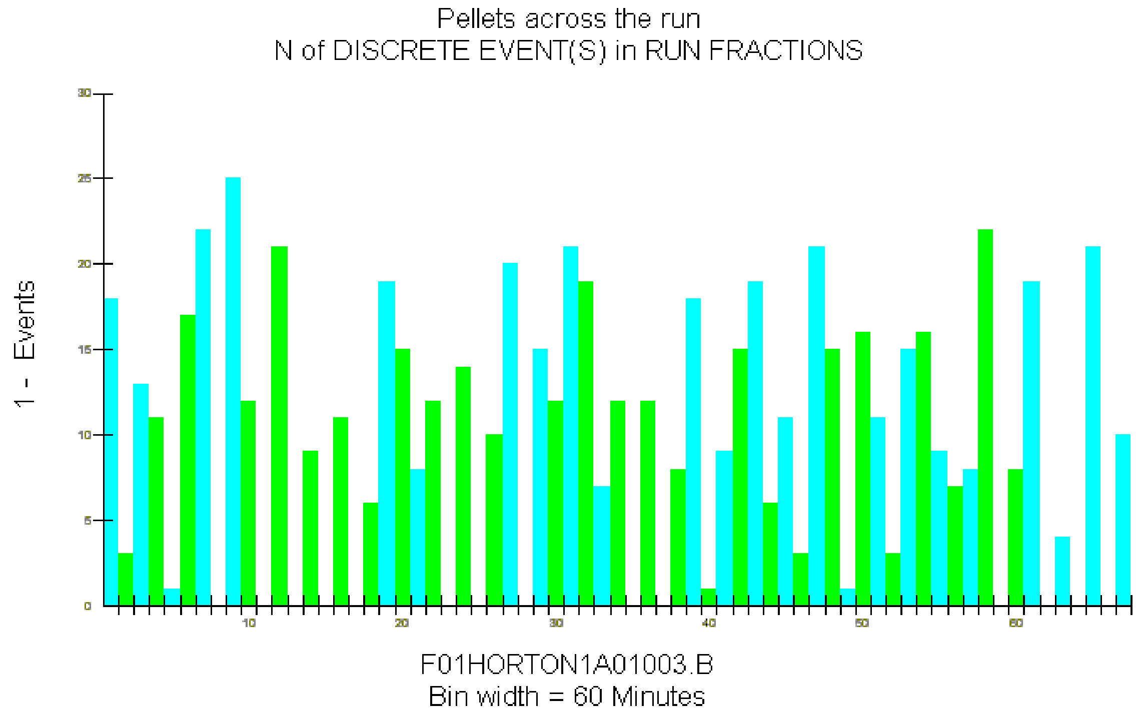

Graphic State --- Project: B Feeding and Drinking - [Create States - Protocol 1]
File Portables Lists Conjoin Help
State List
RDY Ready state
FIN Finished state
S1 IF PELLET REMOVED
S2 DELIVER ANOTHER
Add Entry Go To
Add Event Go To
Add Time Go To
New State
Copy This State
Delete This State
Resolve States
Globals
Finished w/States
State S1 - IF PELLET REMOVED
CUES FEEDERS SPARES AUX PRG
NULL H LITE TONE
R IF 1 2 - Pellet removal GO TO S2


Graphic State --- Project: B Feeding and Drinking - [Create States - Protocol 1]
File Portables Lists Conjoin Help
State List
RDY Ready state
FIN Finished state
S1 IF PELLET REMOVED
S2 DELIVER ANOTHER
Add Entry Go To
Add Event Go To
Add Time Go To
New State
Copy This State
Delete This State
Resolve States
Globals
Finished w/States
State S2 - DELIVER ANOTHER
CUES FEEDERS SPARES AUX PRG
NULL H LITE TONE
R AFTER 1 Units GO TO S1

We can add a Global time GOTO to control the length of the run. For a 23 ½ hour run, it would look like the example below.

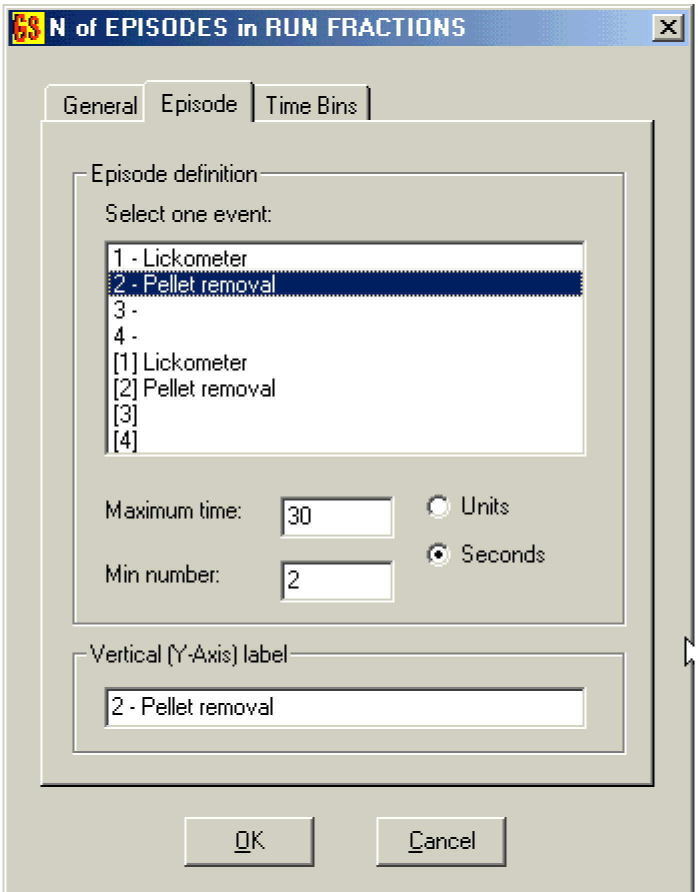


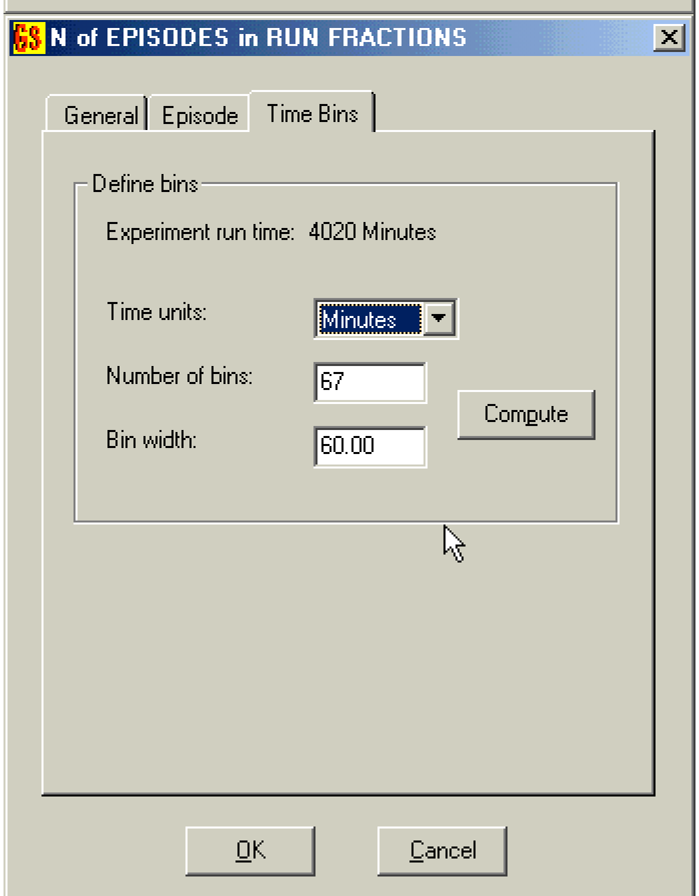
You might be interested in the total number of water licks and pellet removals for the entire period, but you are probably more interested in the distribution of these across the run. I tested a new feeder trough prototype here at Coulbourn with a 67 hour run (4020 minutes) and I have used the data from that run for the analysis examples that follow. The same kinds of analysis would, of course, be applicable to shorter runs as well. First, I created an analysis element using the “N of DISCRETE EVENT(S) in RUN FRACTIONS” element type. I specified the “pellet removal” event and bins of 60 minutes; this produced the following graph.



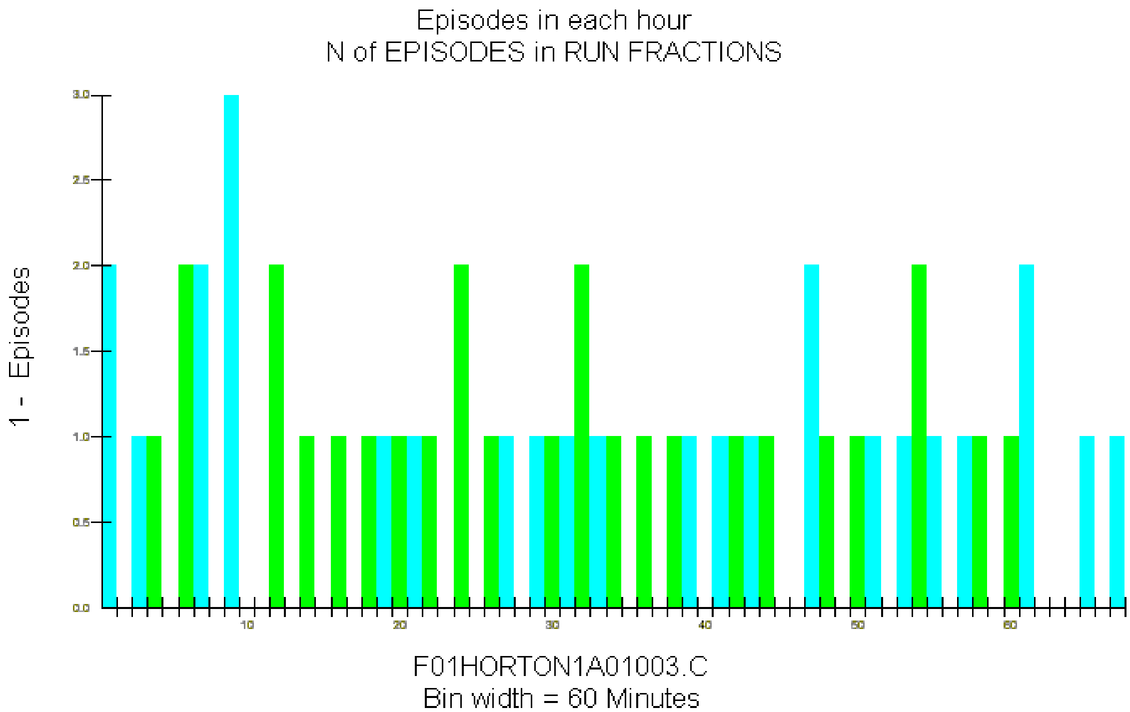
Each bar on the graph represents the number of pellets removed (and presumed eaten) in one hour. I started the run at 3:00 PM on a Friday and ended it at 10:00 AM on Monday morning. The mouse used in this test was more of a pet than an experimental animal and his circadian cycle was not typical. However, you can see that there are bursts of feeding interspersed with periods of no feeding.

We can get a different view of the data by defining feeding "episodes." Using the analysis element type "N of EPISODES in RUN FRACTIONS", I defined an episode as being two or more pellet removals that occurred with no more than 30 seconds between any two. The screen shown below is where these parameters were entered. The second screen shows how I then asked to see the number of such episodes that occurred in each hour of the run.


N of EPISODES in RUN FRACTIONS
General
Episode
Time Bins
Episode definition
Select one event:
1 - Lickometer
2 - Pellet removal
3 -
4 -
[1] Lickometer
[2] Pellet removal
[3]
[4]
Maximum time:
30
Units
Seconds
Min number:
2
Vertical (Y-Axis) label
2 - Pellet removal
OK
Cancel

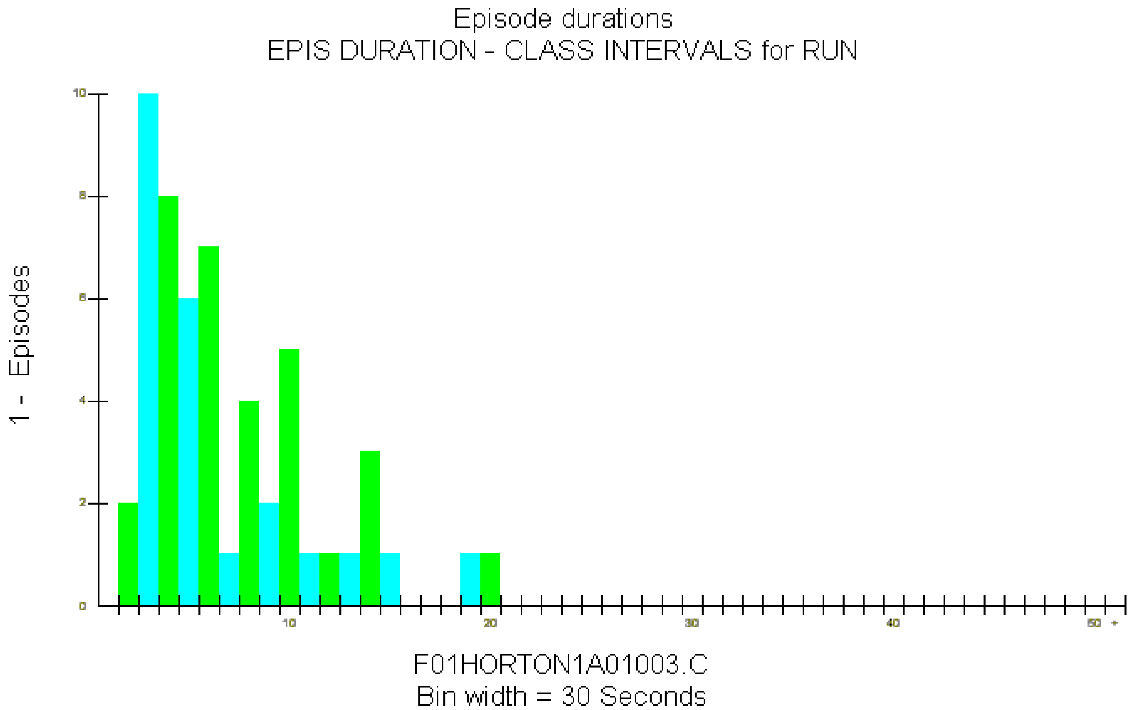

N of EPISODES in RUN FRACTIONS
General
Episode
Time Bins
Define bins
Experiment run time: 4020 Minutes
Time units:
Minutes
Number of bins:
67
Compute
Bin width:
60.00
OK
Cancel

The result of this analysis is shown below.



This graph shows us there were many long episodes of feeding rather than a series of short ones; during all but one of the hour periods, only one or two episodes occurred. This analysis does *not* tell us how long an episode was nor how many pellets were removed during each episode, but it does tell us how “steadily” the animal was eating. As long as the mouse was eating without a break of longer than 30 seconds, this was counted as a single episode. An episode that began in one hour and continued into a second was counted *only* in the first time period.

How long were the feeding episodes? What was the distribution of durations? To find out, I used the element type “EPIS DURATION – CLASS INTERVALS for RUN” defining an episode with the same parameters.



No episode lasted less than 30 seconds (this is a consequence of the definition of an episode in this case), the modal length was 90 seconds, and the longest was 600 seconds (10 minutes).

There is no limit to the number of different analyses that could be made with different definitions of an episode. These same analyses could be specified for the drinking data as well, of course.

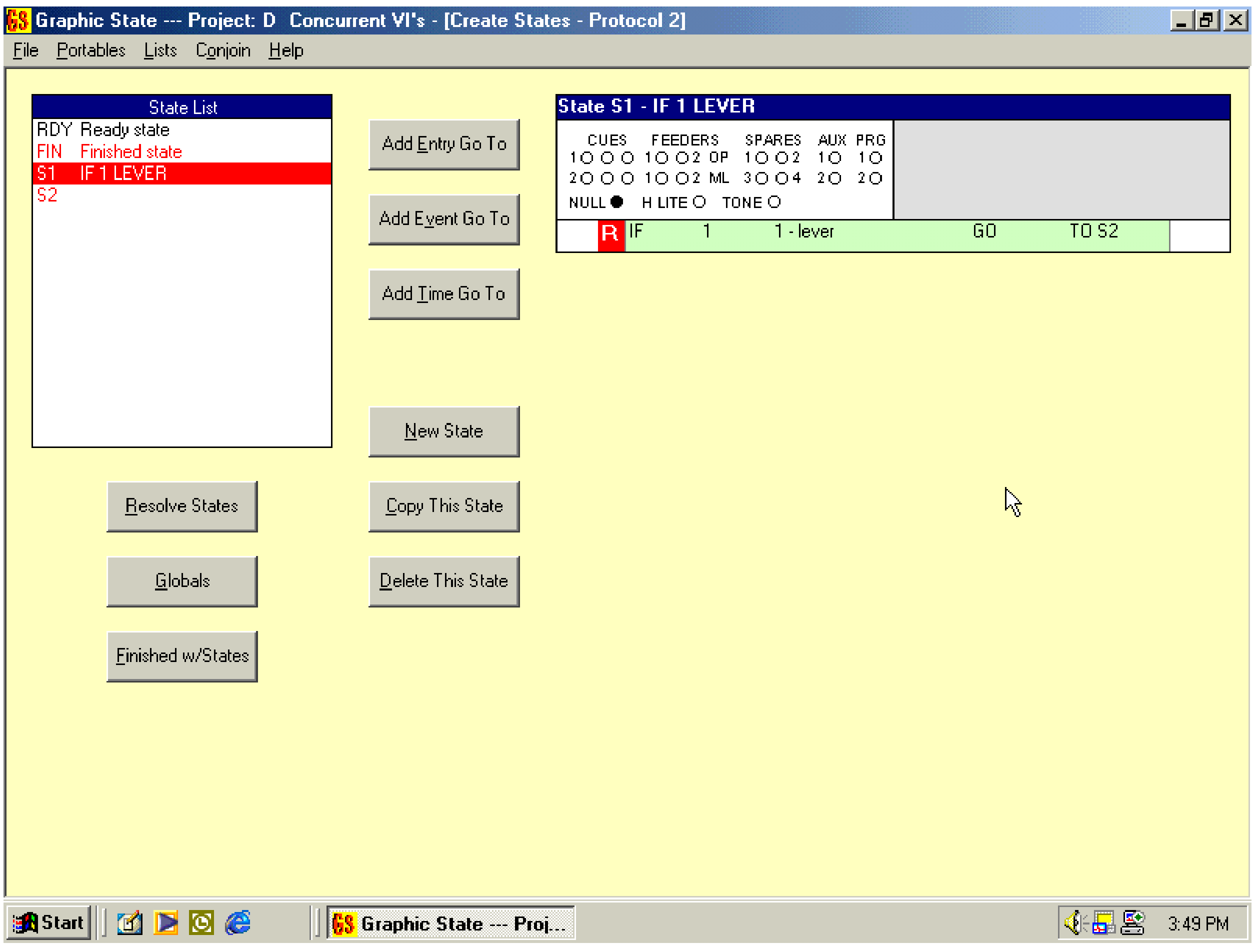
Concurrent schedules

Concurrent schedules pose an interesting problem for state notation languages. How do we superimpose one schedule on top of another and keep all the timers straight?

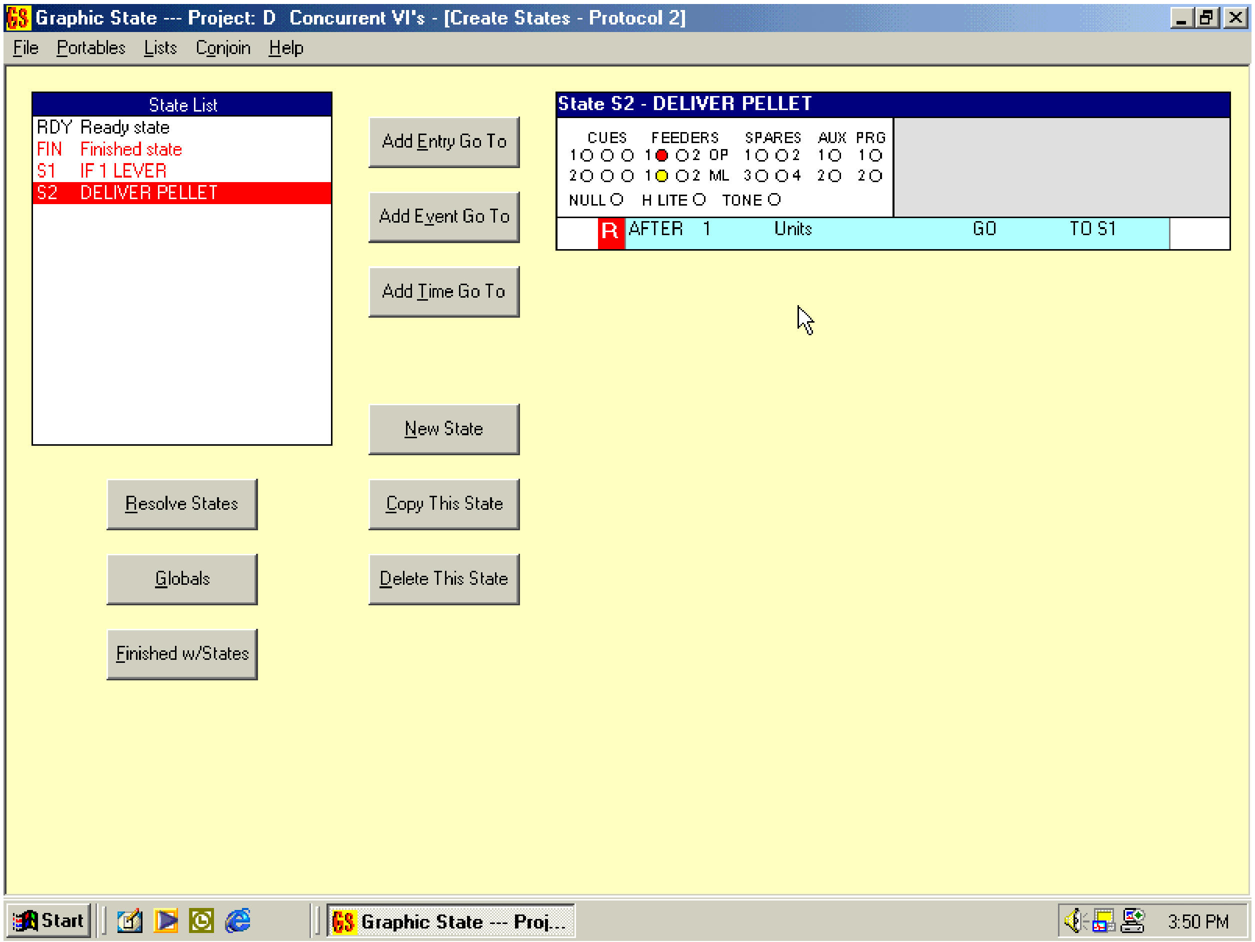
As an example, consider a schedule where 15-second periods of continuous reinforcement (one lever press results in delivery of one food pellet) are separated by variable intervals of extinction (during which no reinforcers are delivered). Then, let's superimpose over this the condition that 10-second periods with a tone sounding are separated by variable time intervals with no tone.

Let's start by ignoring the tone/no tone schedule and dealing just with the VI and CRF combination.

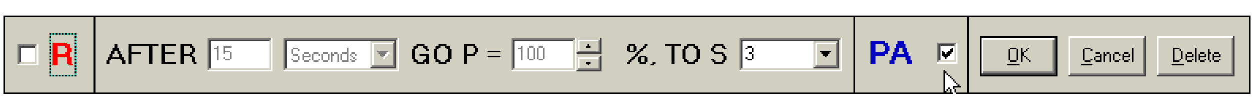
In the first state, we'll check for the lever press response and GOTO a reinforcer delivery state.



The second state will deliver one pellet and immediately return to S1.

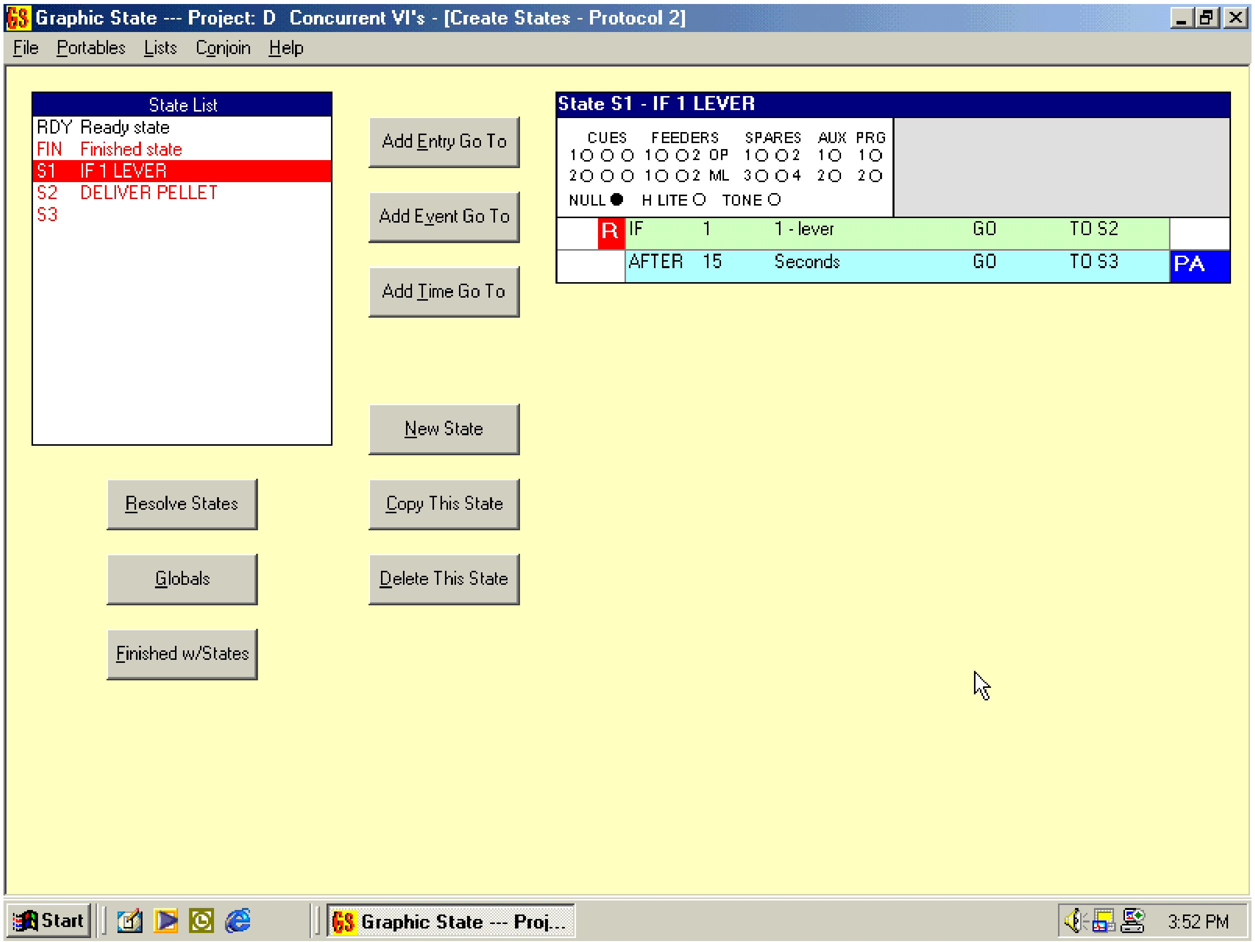


However, we want this pattern to be active for only 15 seconds before we enter a variable length extinction period. We want a *single* timer to count down the 15 seconds no matter how much of the time is spent in S1 and how much is spent in S2. We will add the AFTER 15 seconds statement to S1 and check the box on the statement that makes it a “portable” statement.



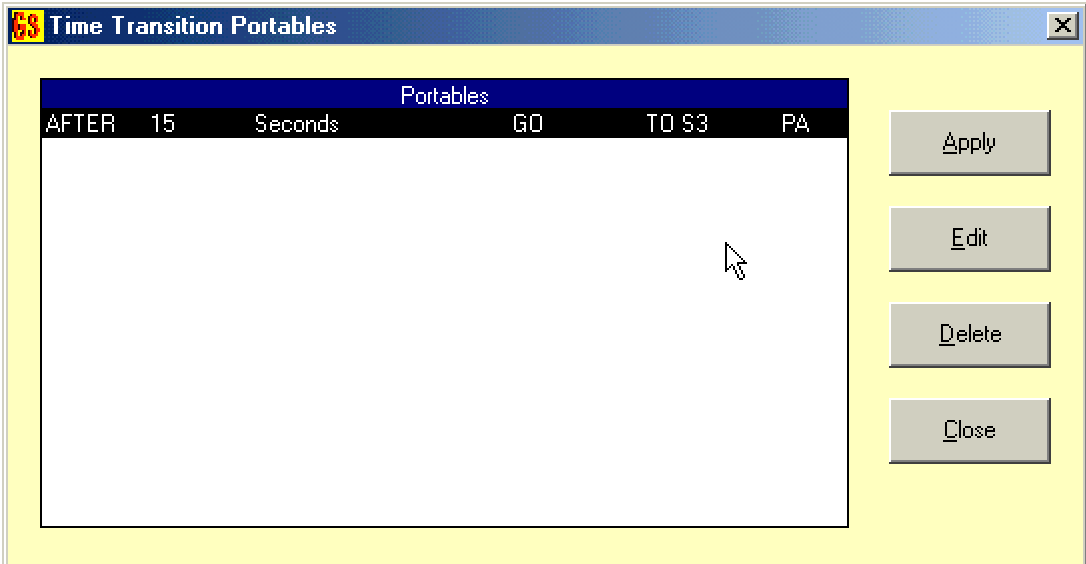
The box is labeled “PA” because it is the first (“A”) portable in this protocol. The subsequent portables we will create will be “PB”, “PC”, etc.

Here’s what it looks like after we add the AFTER statement.

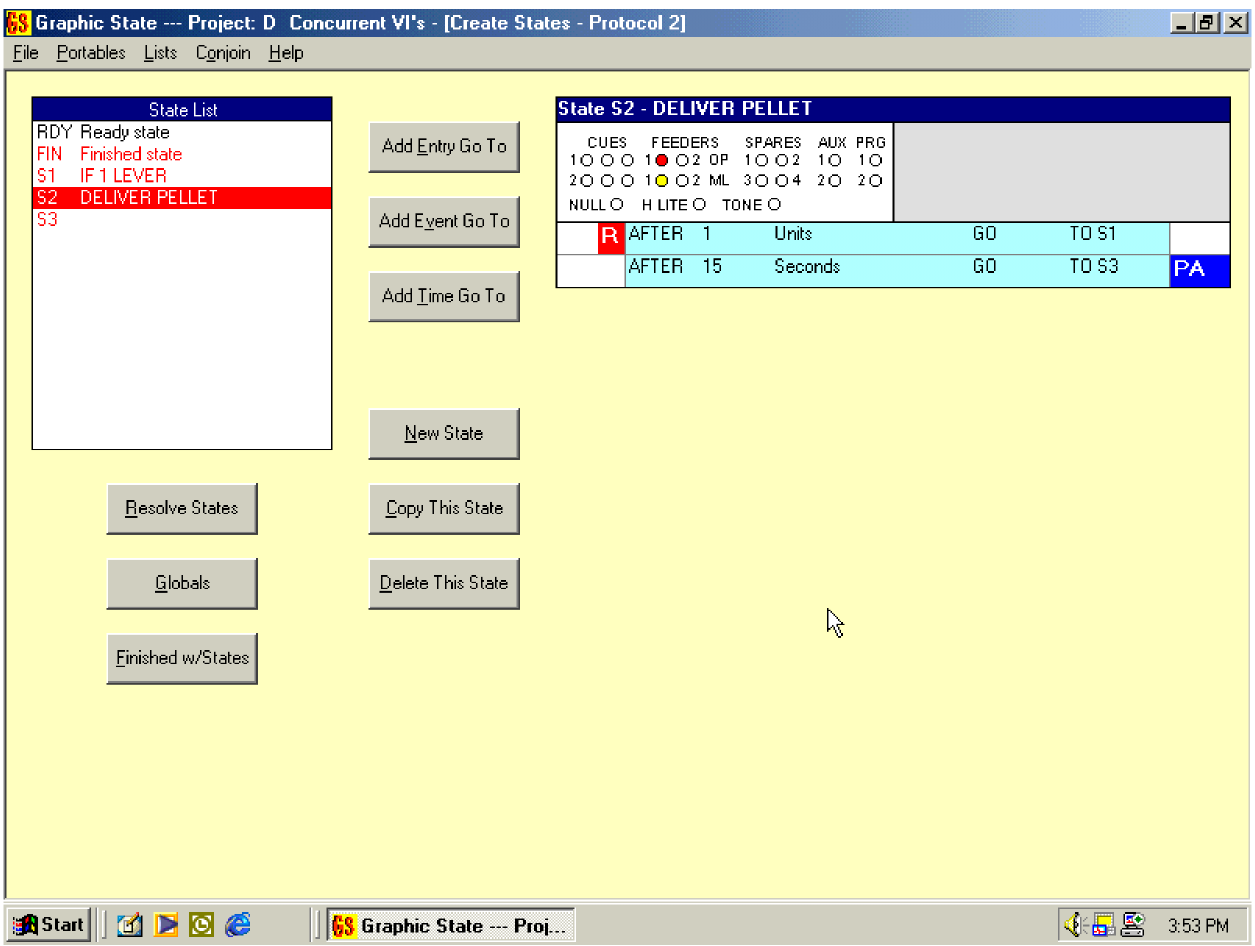


In S2, even though the program will be there for very brief periods, we want to add the same test. The 15 seconds could end while the program is in S2. Even more importantly, 10 pellet deliveries (i.e. 10 visits to S2) would collectively take a full second and we want this time to count towards the 15 seconds.

To add the portable time test to S2, first select S2 from the State List and then click on "Portables" on the command line and then click on "Time Portables". You will see a screen that lists the currently available time portables; there will be only one at this time.



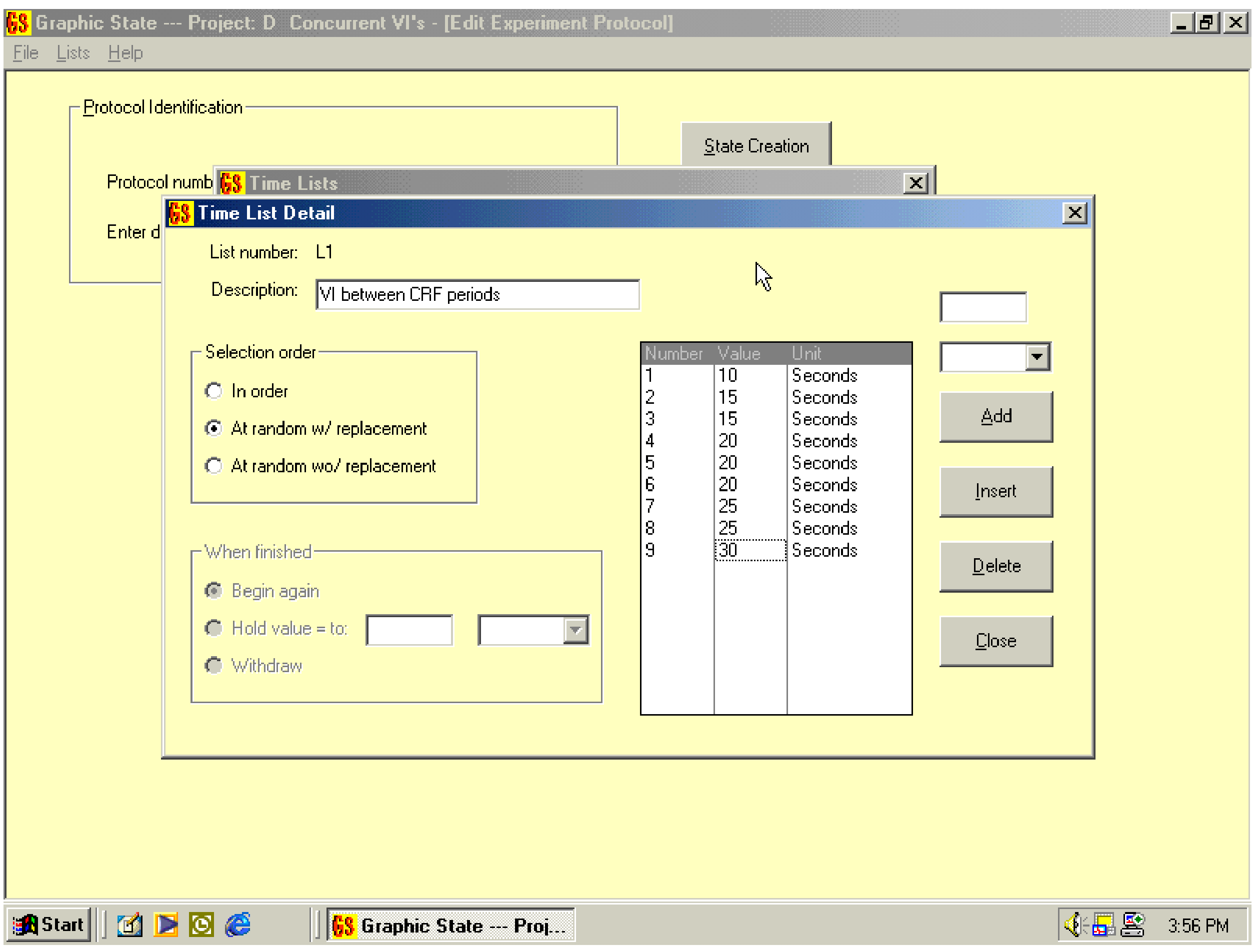
Click the "Apply" button and the statement will be added to S2.



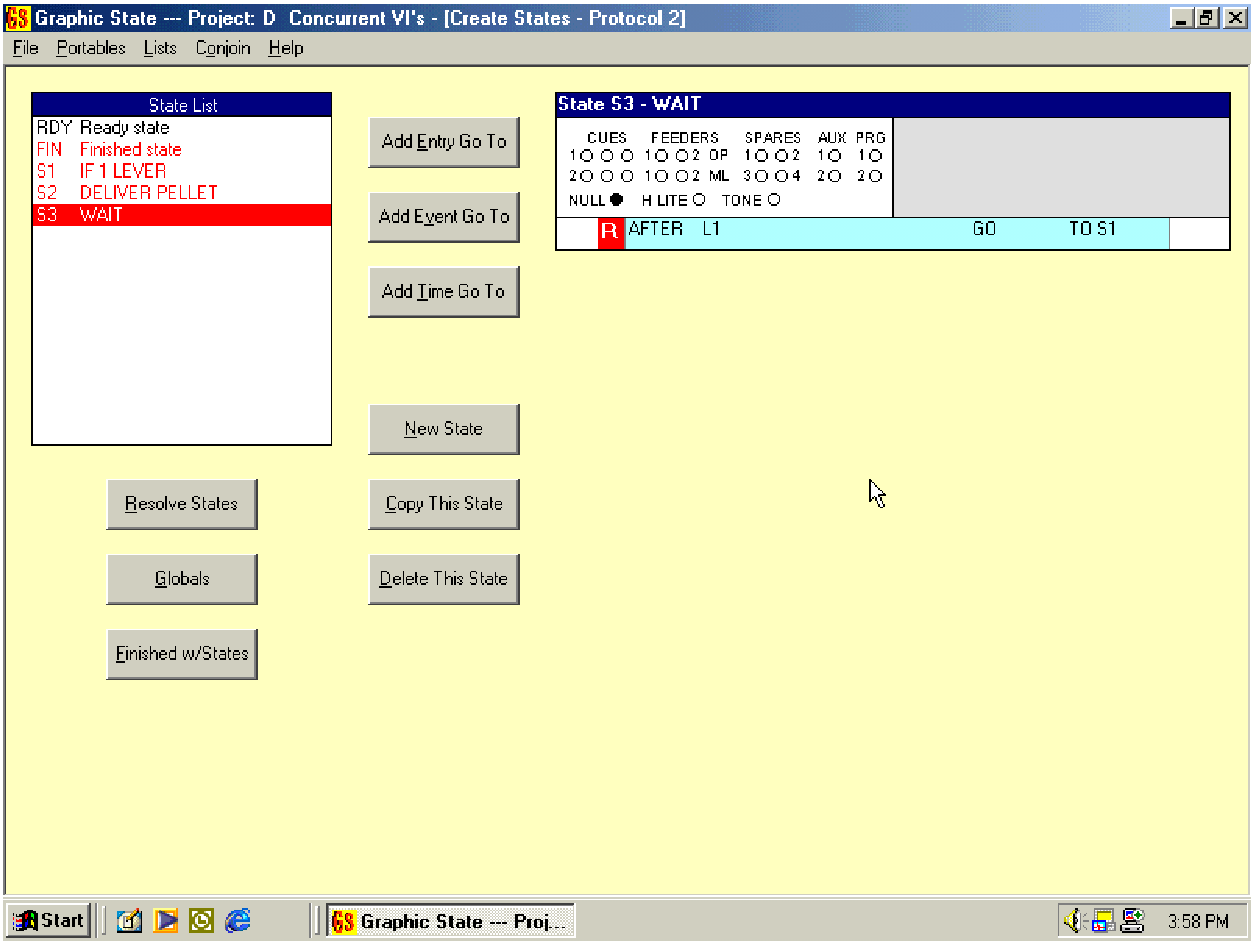
What you do NOT want to do instead is to use the “Add Time Go To” button here in S2. If you do, you may create a NEW portable (it would be “PB”) instead of using the existing one.

We want the protocol to stay in S3 (the non-reinforcement phase) for a variable length of time. We’ll use a time list for this, so let’s create the list next.

Use “Lists” from the command line., select “Database level lists” and then “Time Transition Parameter Lists” The sample below implements a VI 20 seconds which is weighted so that the most common time will be 20 seconds, the next most common times will be either 15 or 25 seconds and the least common times will be 10 or 30 seconds. This is controlled by the number of times each time interval appears in the list. (As I have said before, there are *many* ways to implement a VI n seconds).



We will use "Add Time Go To" and reference the list we just made.

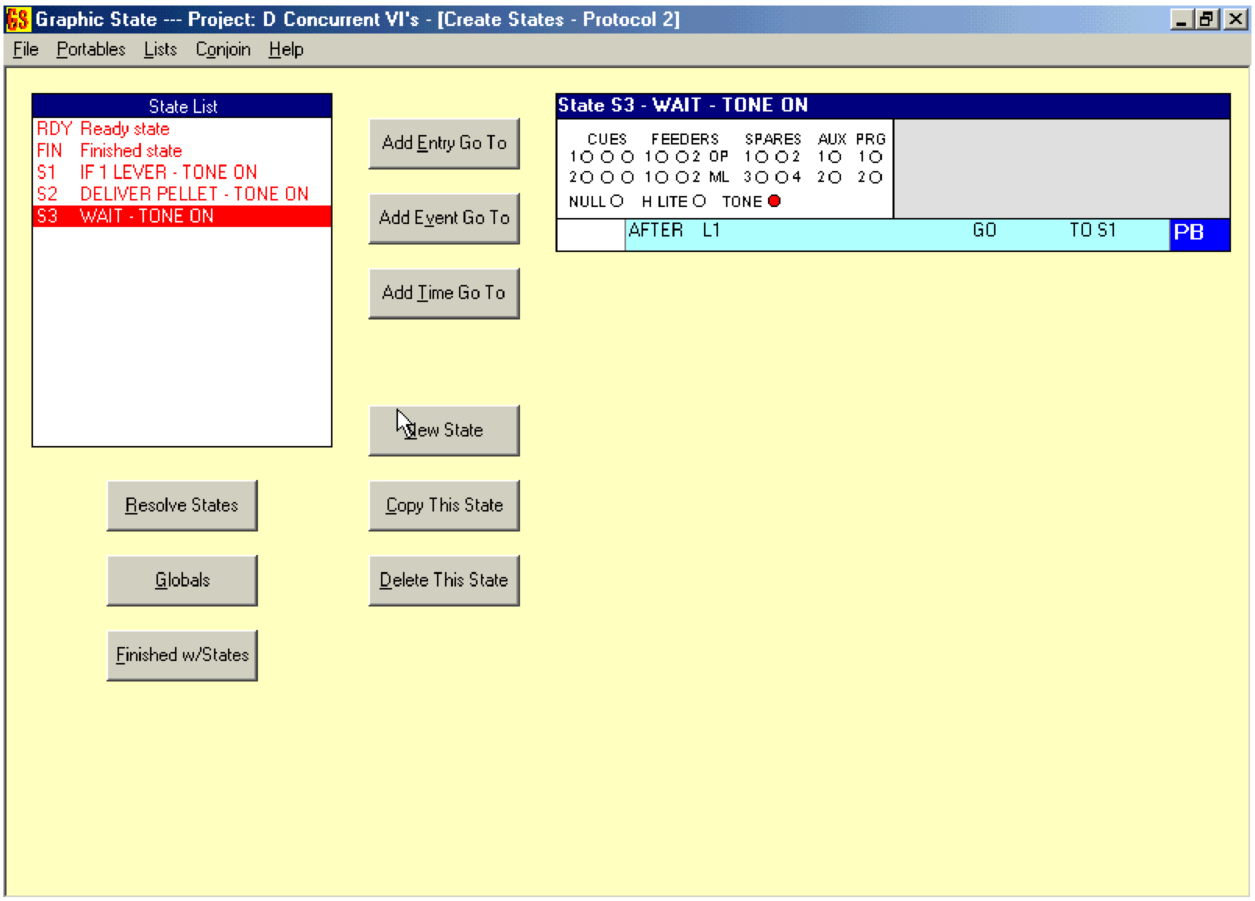


Now we have a functional protocol (if we specified a FIN condition) that implements the periods of CRF interspersed with variable periods of extinction. But, we want to superimpose fixed-length periods with a tone sounding interspersed with variable-length periods with no tone.

To do this we will set up another three states that parallel the first three. In one set, the tone will be sounding; in the other set, the tone will be off. Then we will toggle back and forth between the appropriate corresponding states based on two portable times (one for the constant 10 seconds with tone and another for the variable interval between tones).

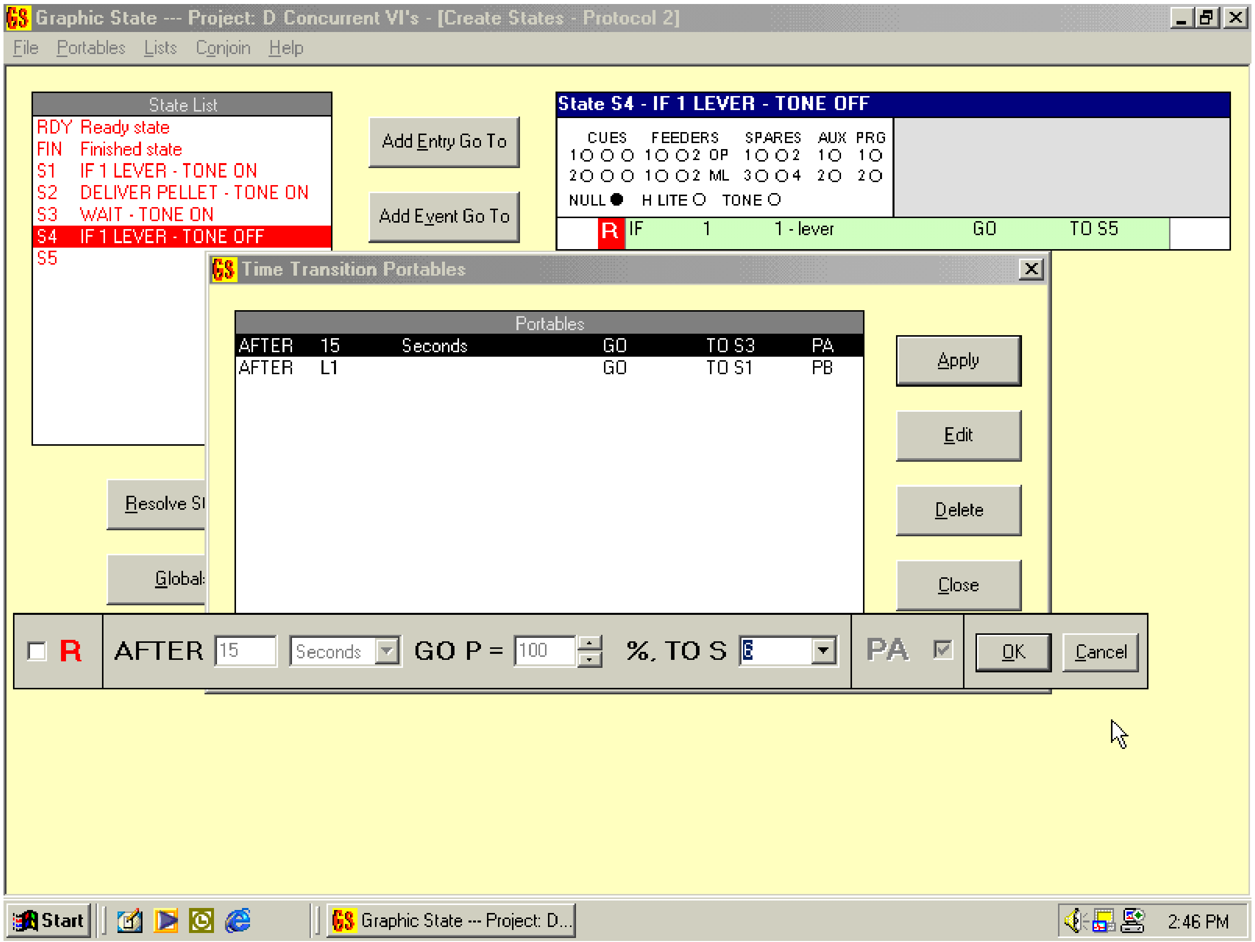
First, click on the “Tone” button in the stimulus graphic for S1, S2 and S3. Change the names of the states at this time to include something to designate that these are the “tone on” states.

We also need to make the AFTER L1 statement in State 3 into a portable because we will be using it in both State 3 and the corresponding State6. It will become “PB”.

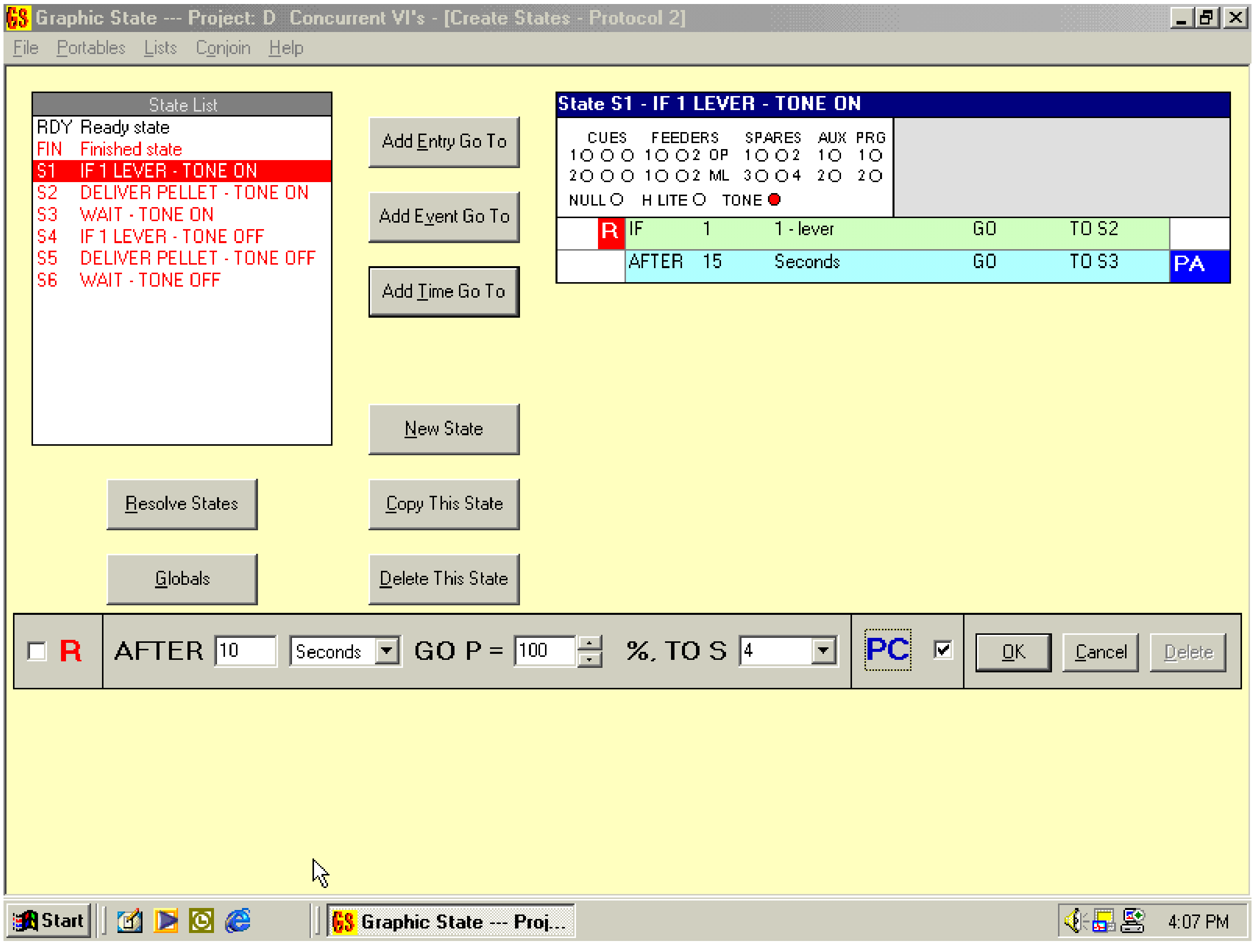


Now, we need to set up the parallel states (S4, S5 and S6). Give them names like “IF 1 LEVER – TONE OFF”, etc. State 4 will test for one lever press to GOTO State 5 and will also include Portable A with the target state modified to be State 6 instead of State

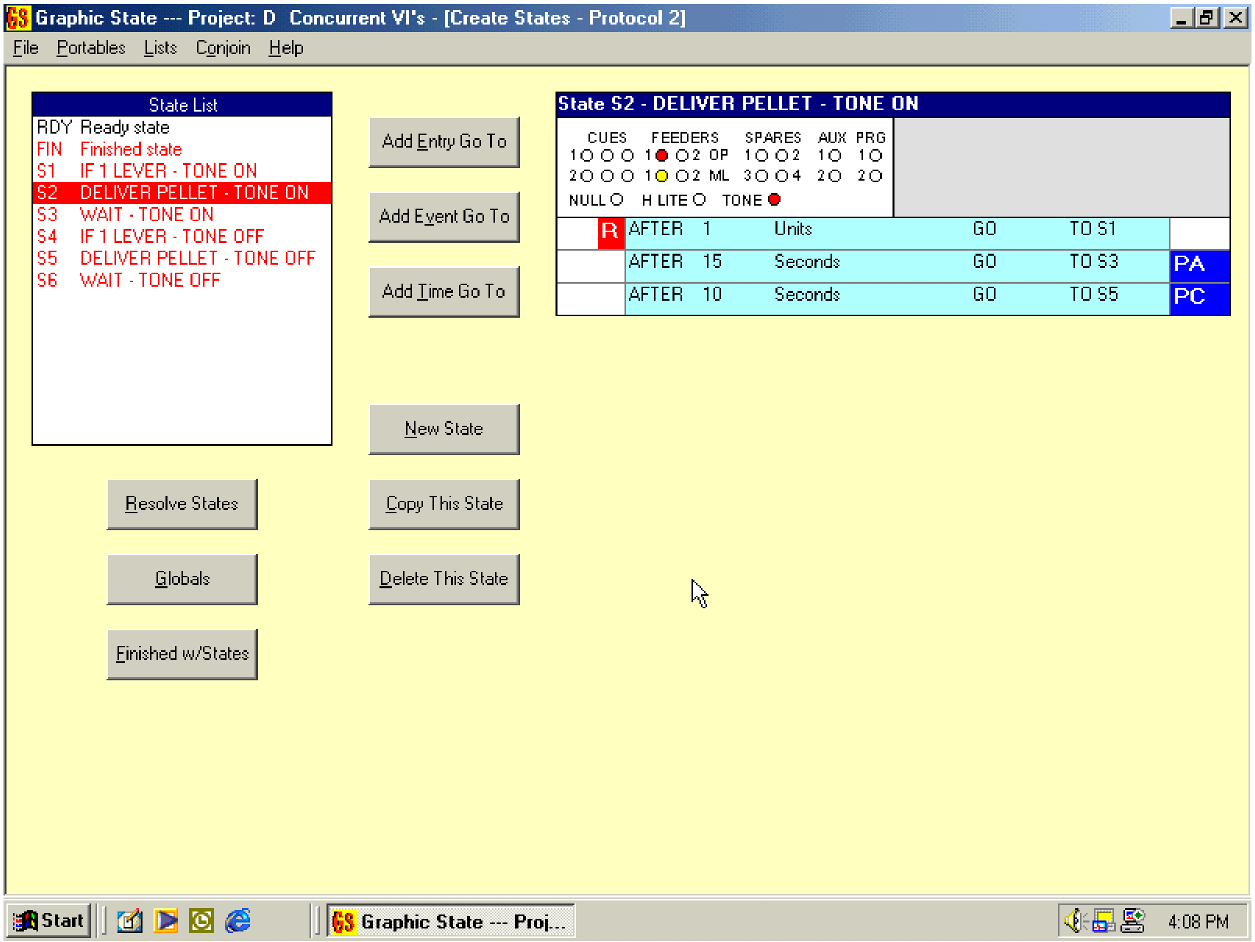
1. (Apply the portable and then type over the GOTO state number.)

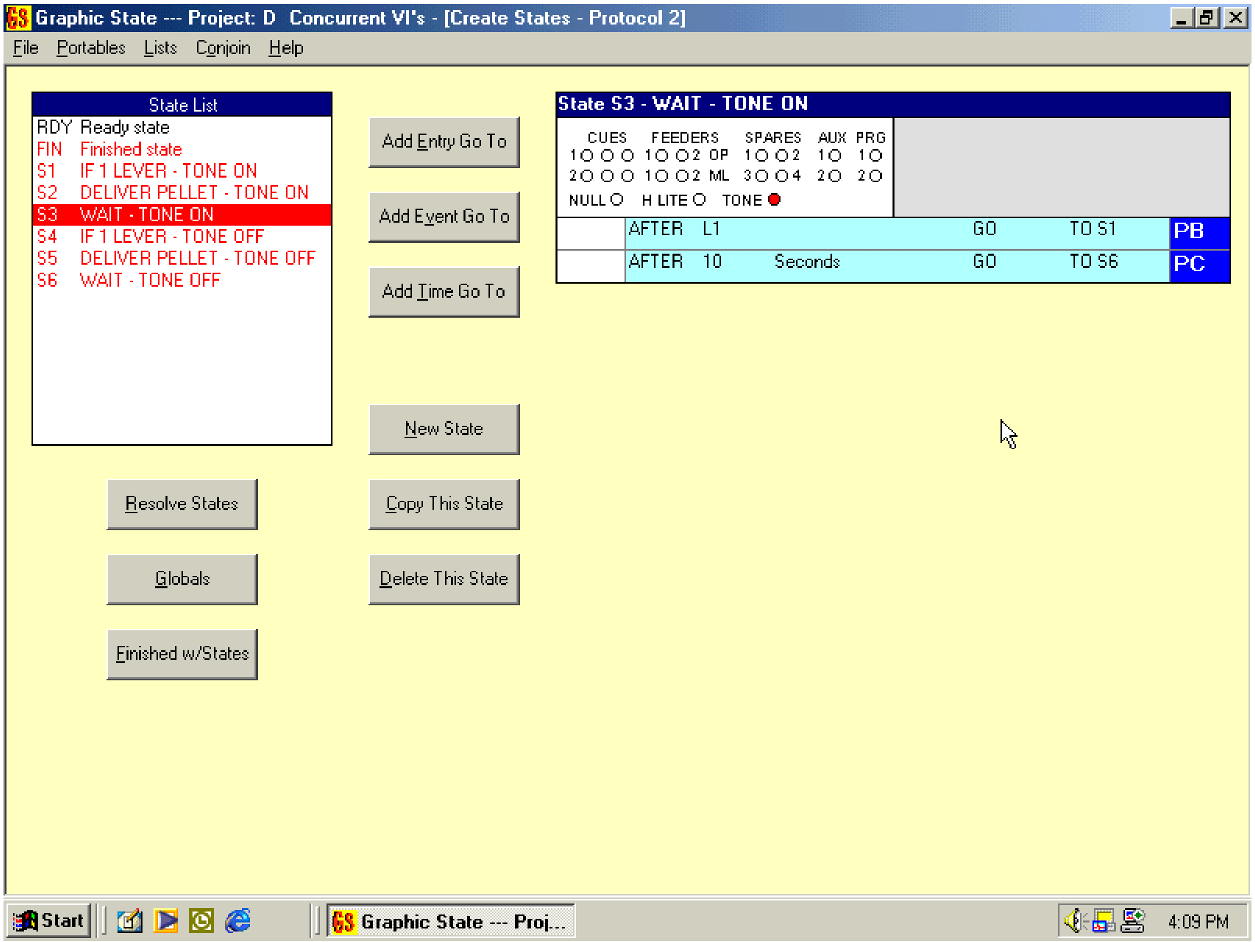


We will now create our third portable (PC) to toggle from the “tone on” states to the “tone off” states after 10 seconds. Create the portable in State 1 to GO TO State 4.

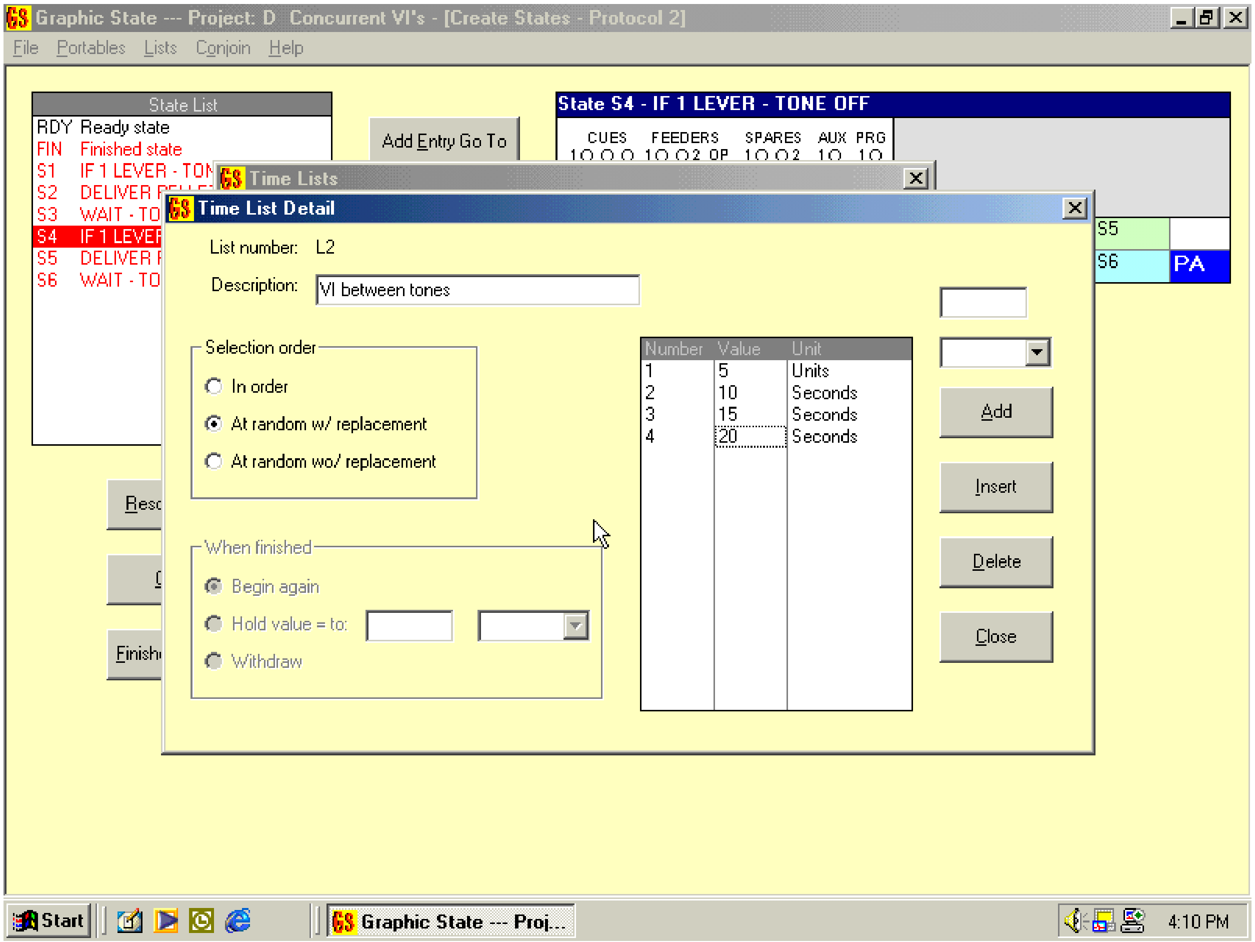


Apply Portable C in State 2 to GO TO State 5 and in State 3 to GO TO State 6.

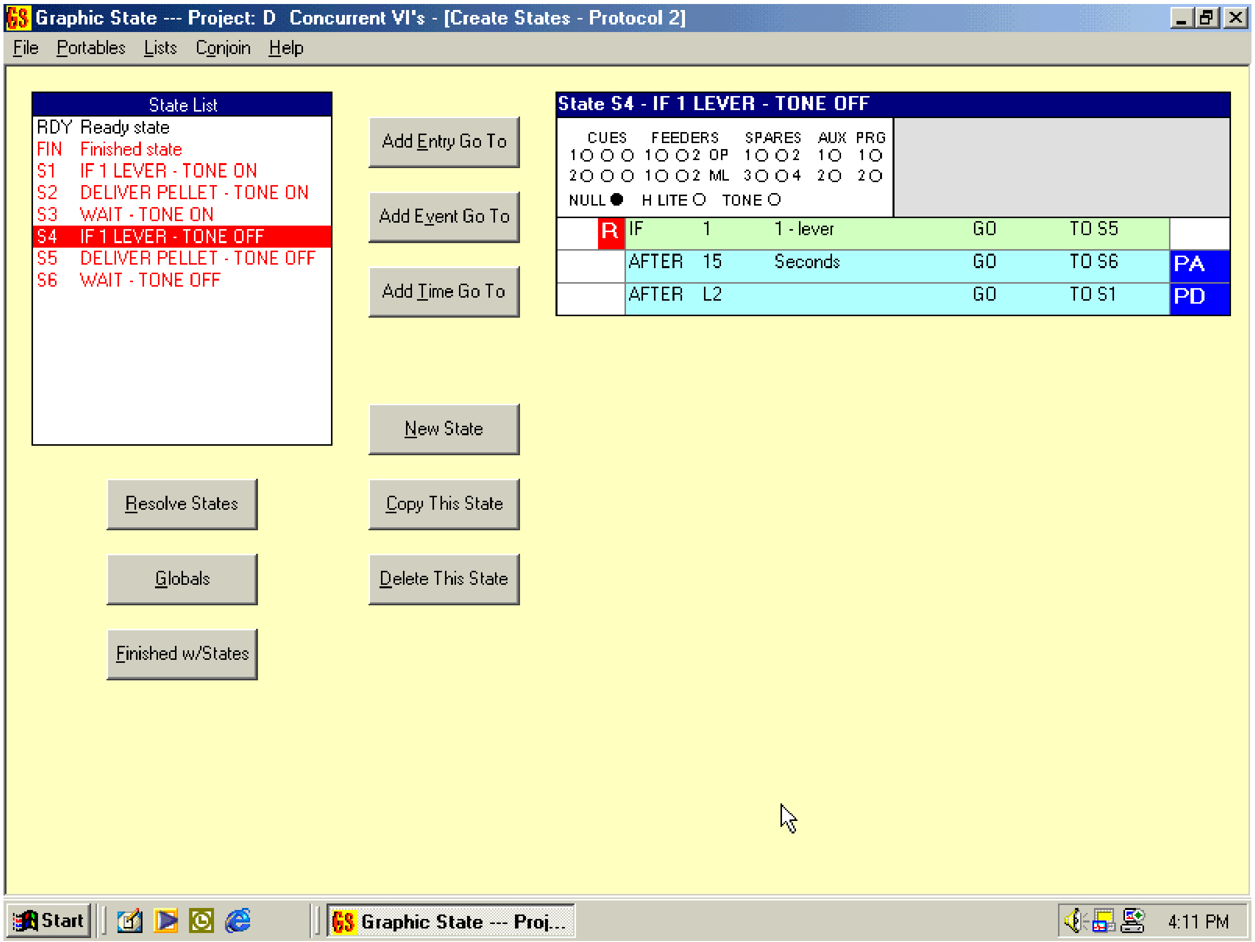

Graphic State --- Project: D Concurrent VI's - [Create States - Protocol 2]
File Portables Lists Conjoin Help
State List
RDY Ready state
FIN Finished state
S1 IF 1 LEVER - TONE ON
S2 DELIVER PELLET - TONE ON
S3 WAIT - TONE ON
S4 IF 1 LEVER - TONE OFF
S5 DELIVER PELLET - TONE OFF
S6 WAIT - TONE OFF
Add Entry Go To
Add Event Go To
Add Time Go To
New State
Copy This State
Delete This State
Resolve States
Globals
Finished w/States
State S2 - DELIVER PELLET - TONE ON
CUES FEEDERS SPARES AUX PRG
OP ML
NULL H LITE TONE
R AFTER 1 Units GO TO S1
AFTER 15 Seconds GO TO S3 PA
AFTER 10 Seconds GO TO S5 PC
Start
Graphic State --- Proj...
4:08 PM



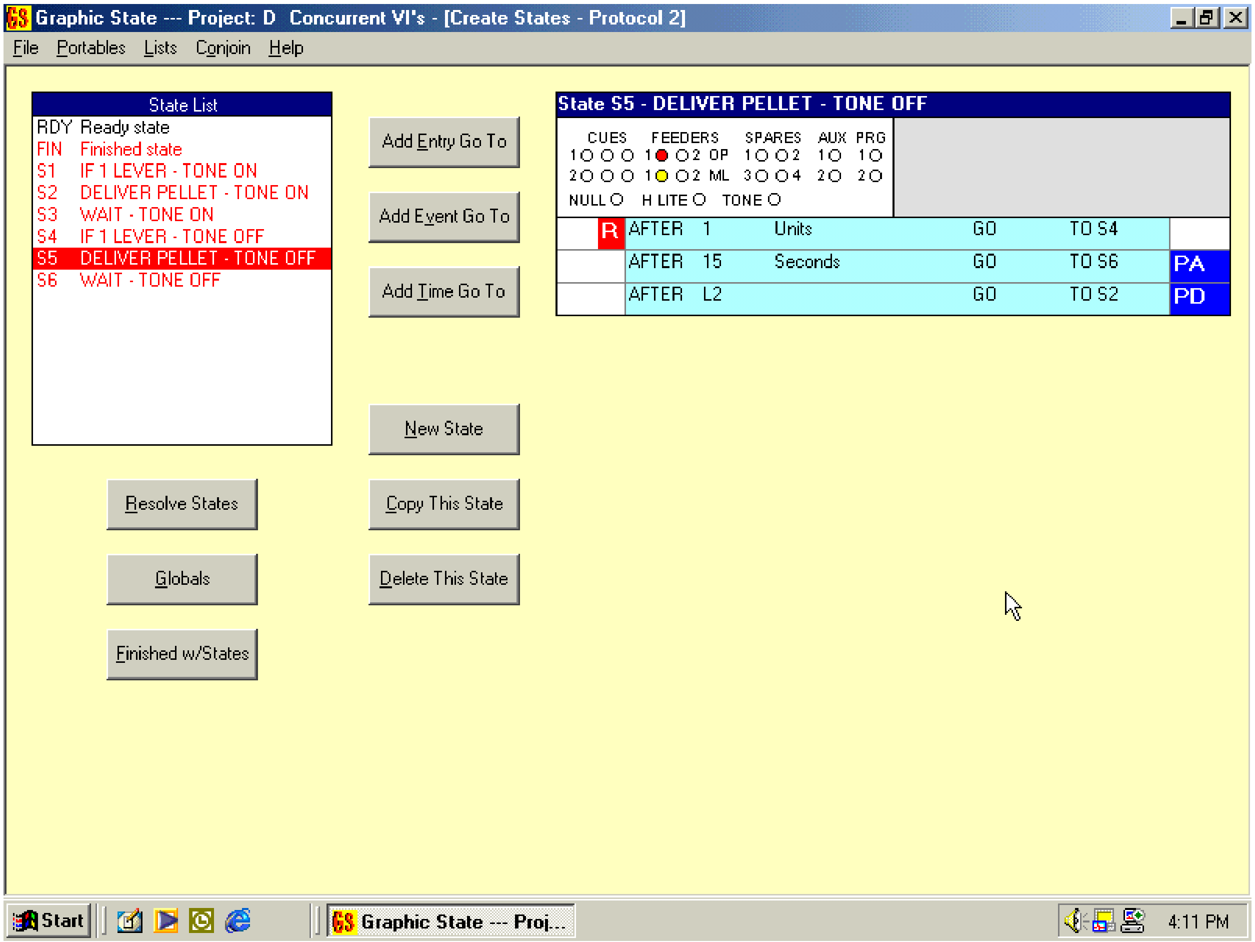
We will use a similar mechanism to toggle back to the "tone on" states from the "tone off" states, except that it will involve a list of times with random selection. In this example, I set up L2 to contain four times (5, 10, 15, and 20 seconds) all equally likely.

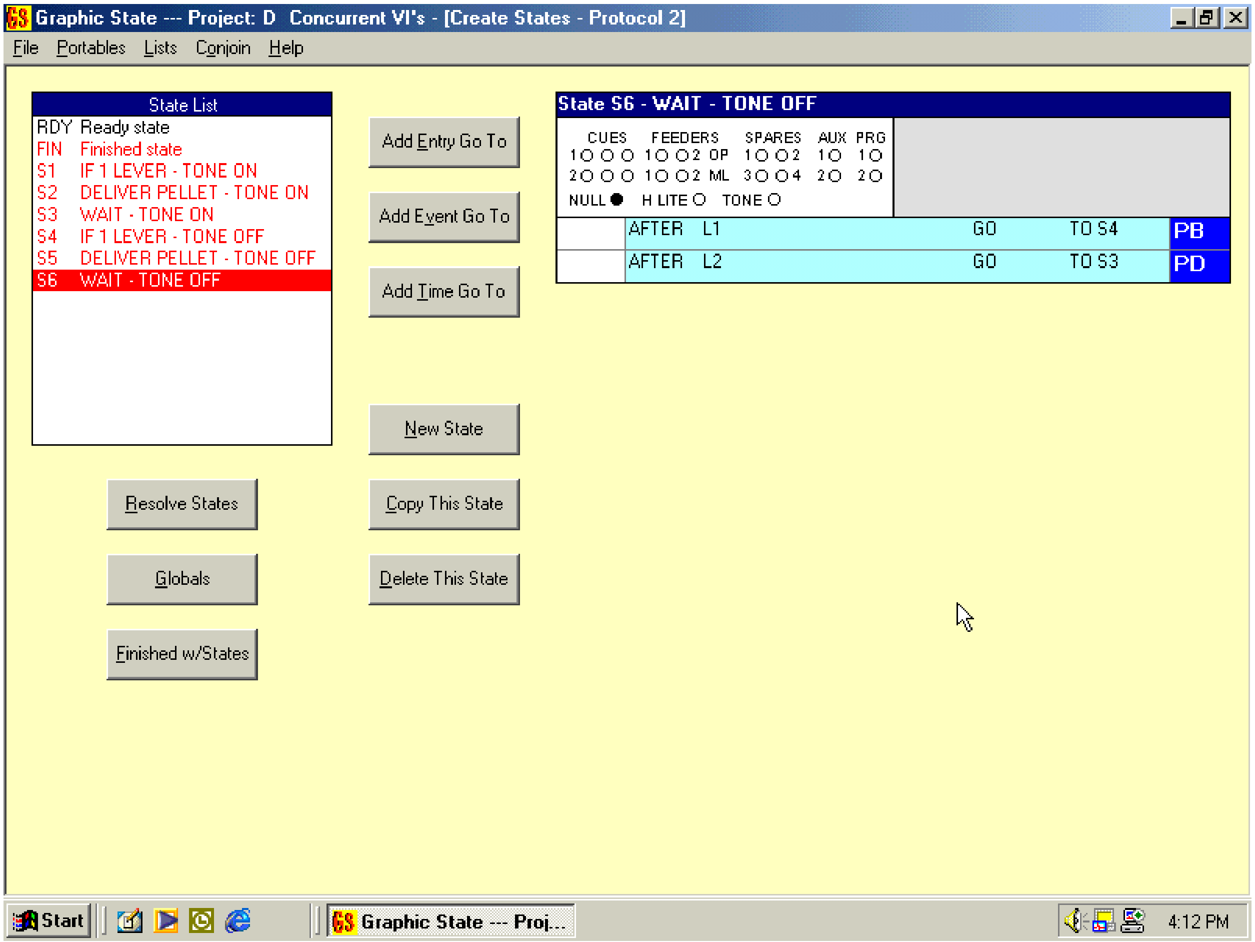


In State 4, we use the list to return to State 1 and thereby define Portable D.

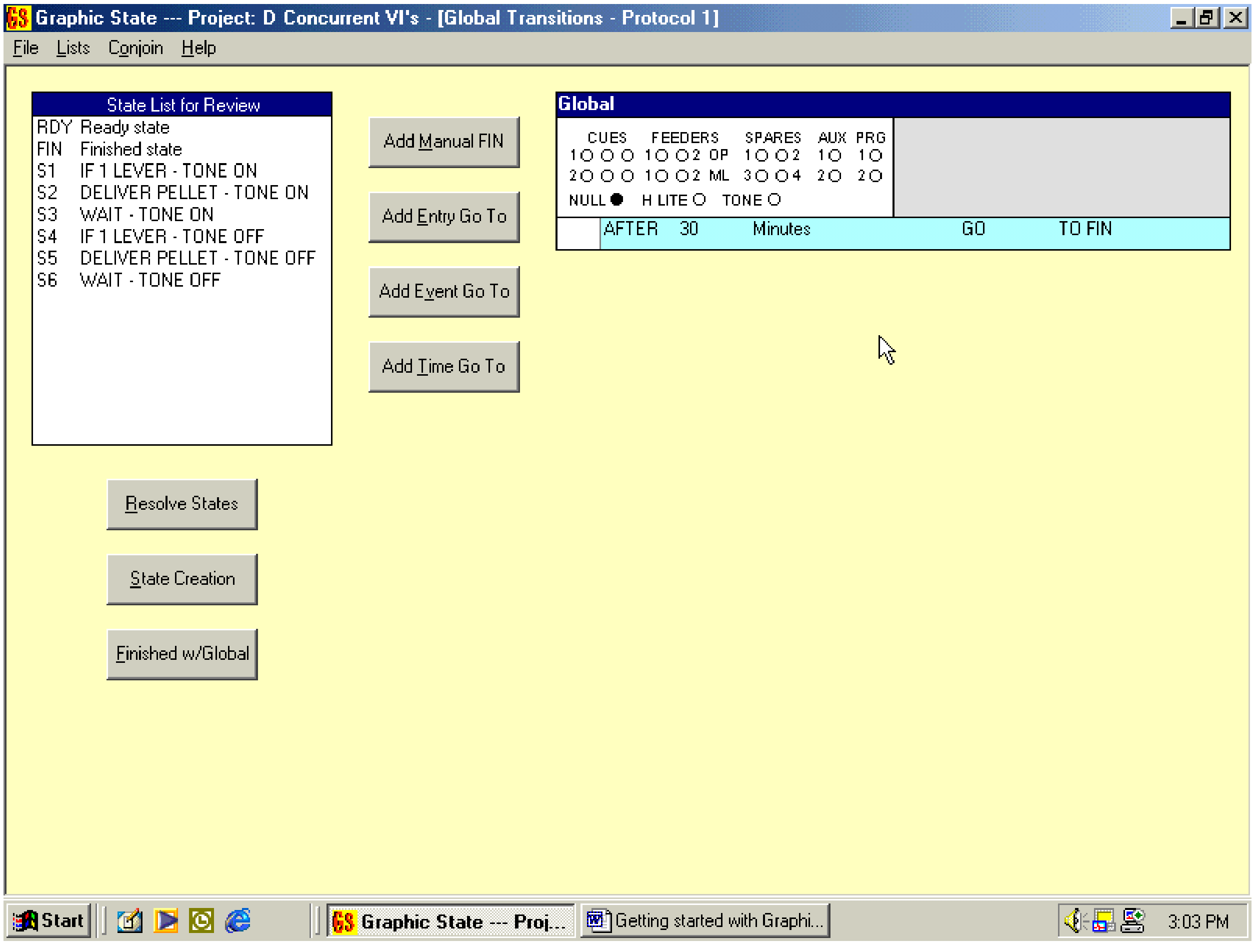


This portable is then applied in State 5 to GO TO State 2 and in State 6 to GO TO State 3.


Graphic State --- Project: D Concurrent VI's - [Create States - Protocol 2]
File Portables Lists Conjoin Help
State List
RDY Ready state
FIN Finished state
S1 IF 1 LEVER - TONE ON
S2 DELIVER PELLET - TONE ON
S3 WAIT - TONE ON
S4 IF 1 LEVER - TONE OFF
S5 DELIVER PELLET - TONE OFF
S6 WAIT - TONE OFF
Add Entry Go To
Add Event Go To
Add Time Go To
New State
Copy This State
Delete This State
Resolve States
Globals
Finished w/States
State S5 - DELIVER PELLET - TONE OFF
CUES FEEDERS SPARES AUX PRG
NULL H LITE TONE
R AFTER 1 Units GO TO S4
AFTER 15 Seconds GO TO S6 PA
AFTER L2 GO TO S2 PD
Start
Graphic State --- Proj...
4:11 PM



The addition of a Global time to GO TO FIN completes this protocol. Click on "Resolve States" and everything will turn black and be ready to run.


Graphic State --- Project: D Concurrent VI's - [Global Transitions - Protocol 1]
File Lists Conjoin Help
State List for Review
RDY Ready state
FIN Finished state
S1 IF 1 LEVER - TONE ON
S2 DELIVER PELLET - TONE ON
S3 WAIT - TONE ON
S4 IF 1 LEVER - TONE OFF
S5 DELIVER PELLET - TONE OFF
S6 WAIT - TONE OFF
Add Manual FIN
Add Entry Go To
Add Event Go To
Add Time Go To
Global
CUES FEEDERS SPARES AUX PRG
NULL H LITE TONE
AFTER 30 Minutes GO TO FIN
Resolve States
State Creation
Finished w/Global
Start
Graphic State --- Proj...
Getting started with Graphi...
3:03 PM

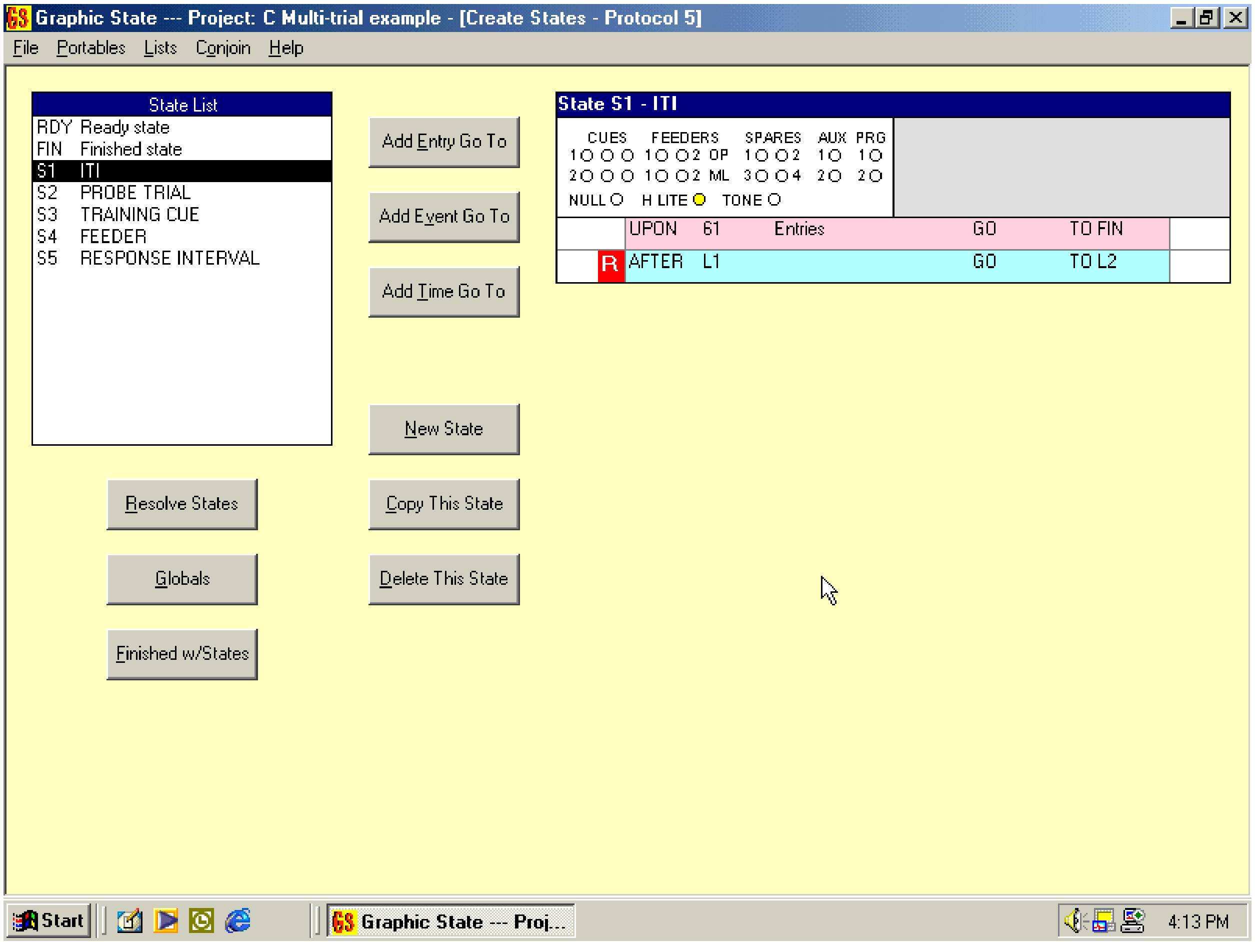
Trial-by-trial data analysis

For the final example in this tutorial manual, I've selected a simple protocol with data analysis that must be implemented *outside of* the Graphic State program using Excel or some other program.

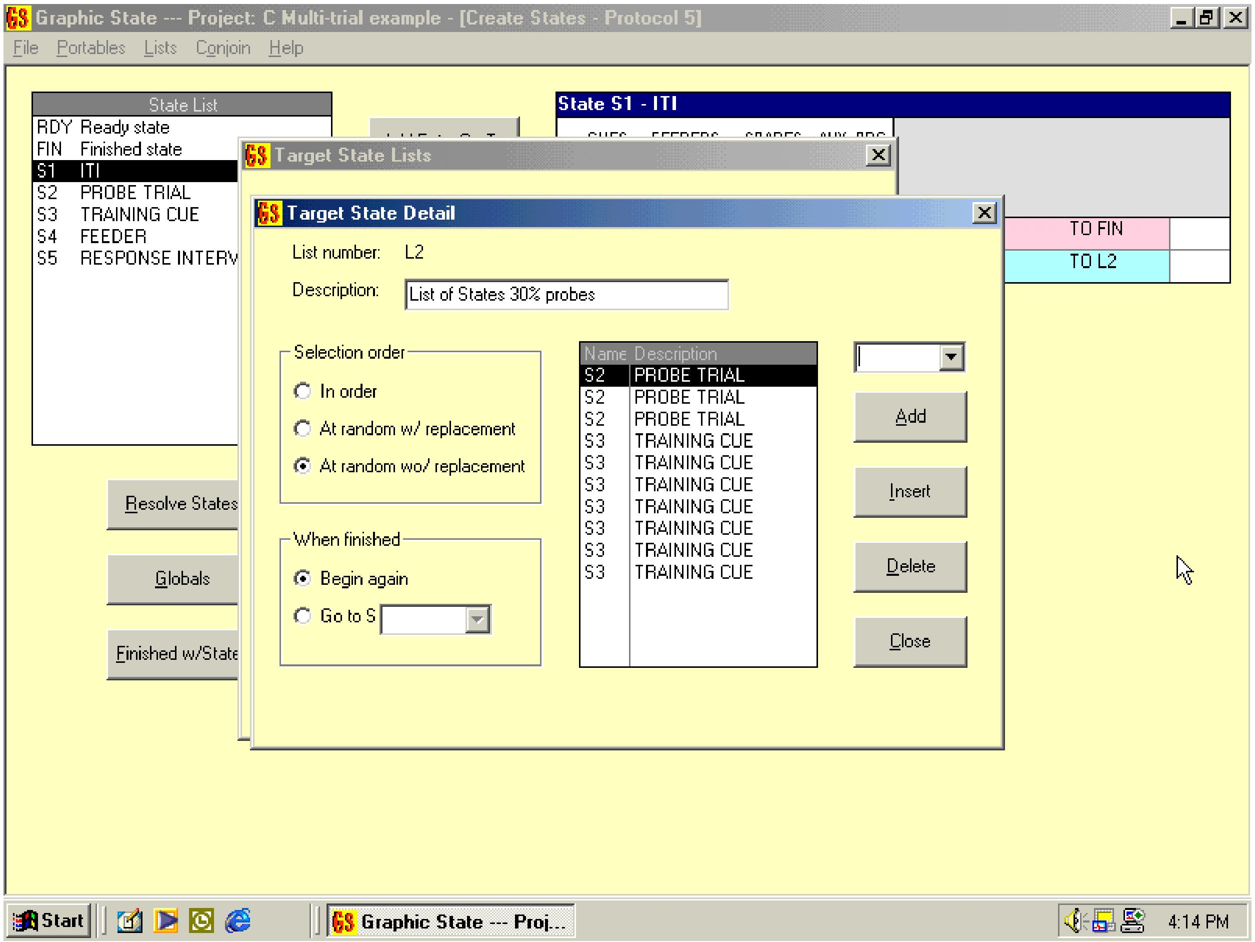
This protocol runs 60 trials in total, 30% of one type ("training") and 70% of a second type ("probe"), in random order. The trials are separated by a variable inter-trial interval. On training trials, a tone is sounded for 20 seconds and then, if a lever press occurs anytime during the next 40 seconds, while the tone continues, a reinforcer is delivered and the trial ends. If there is no lever press, the trial ends after the 40 seconds with no reinforcer delivery. On probe trials, the tone sounds for the full 60 seconds with no reinforcer deliveries.

State 1 controls the length of each inter-trial interval, the random selection of trial types and the termination of the run. We want to run 60 trials and State 1 is the starting point for each trial. I have used an UPON statement to test for the 61st attempt to enter State 1; this attempt is the signal to GO TO FIN.

Two lists are used to control 1) which state to GO TO next and 2) after how much time.

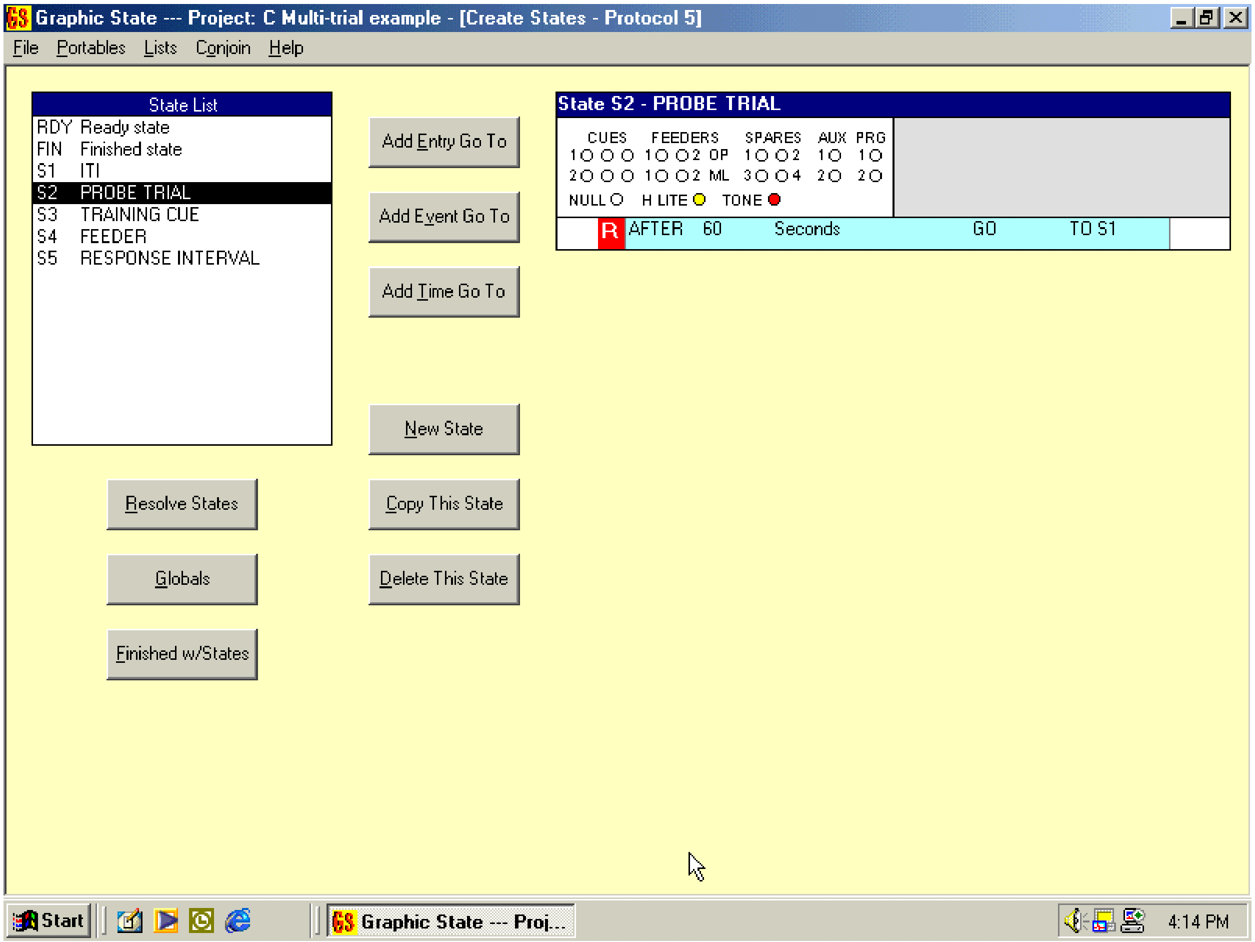


L1 is a list of times like those we have created before. L2 is a "Target State List" found under "Protocol Level Lists" under "Lists".

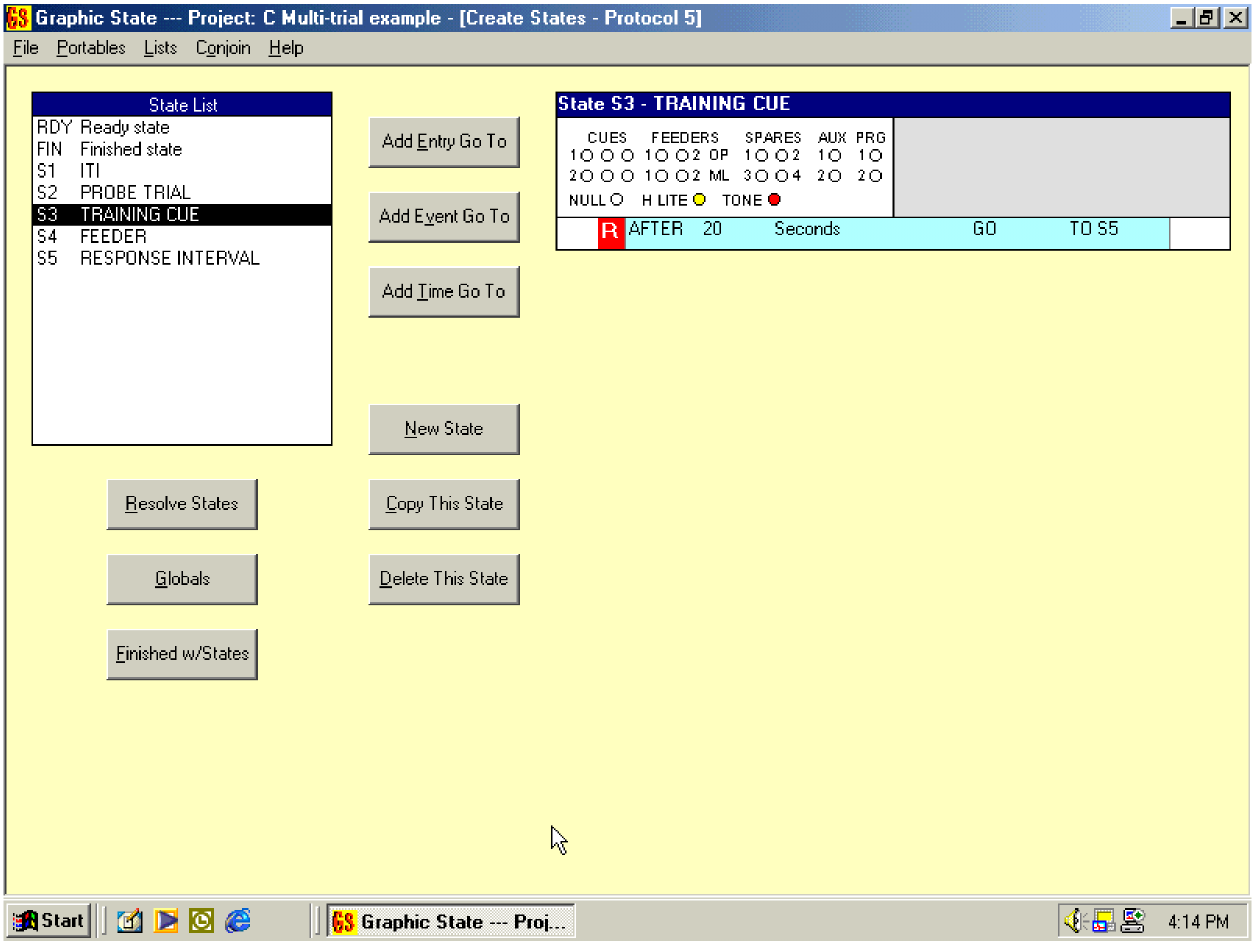


Notice that S2 is in the list three times and S3 is in the list seven times. This, along with the selection order “At random wo/ replacement”, guarantees that in every set of ten trials, there will be exactly three probe trials and seven training trials.

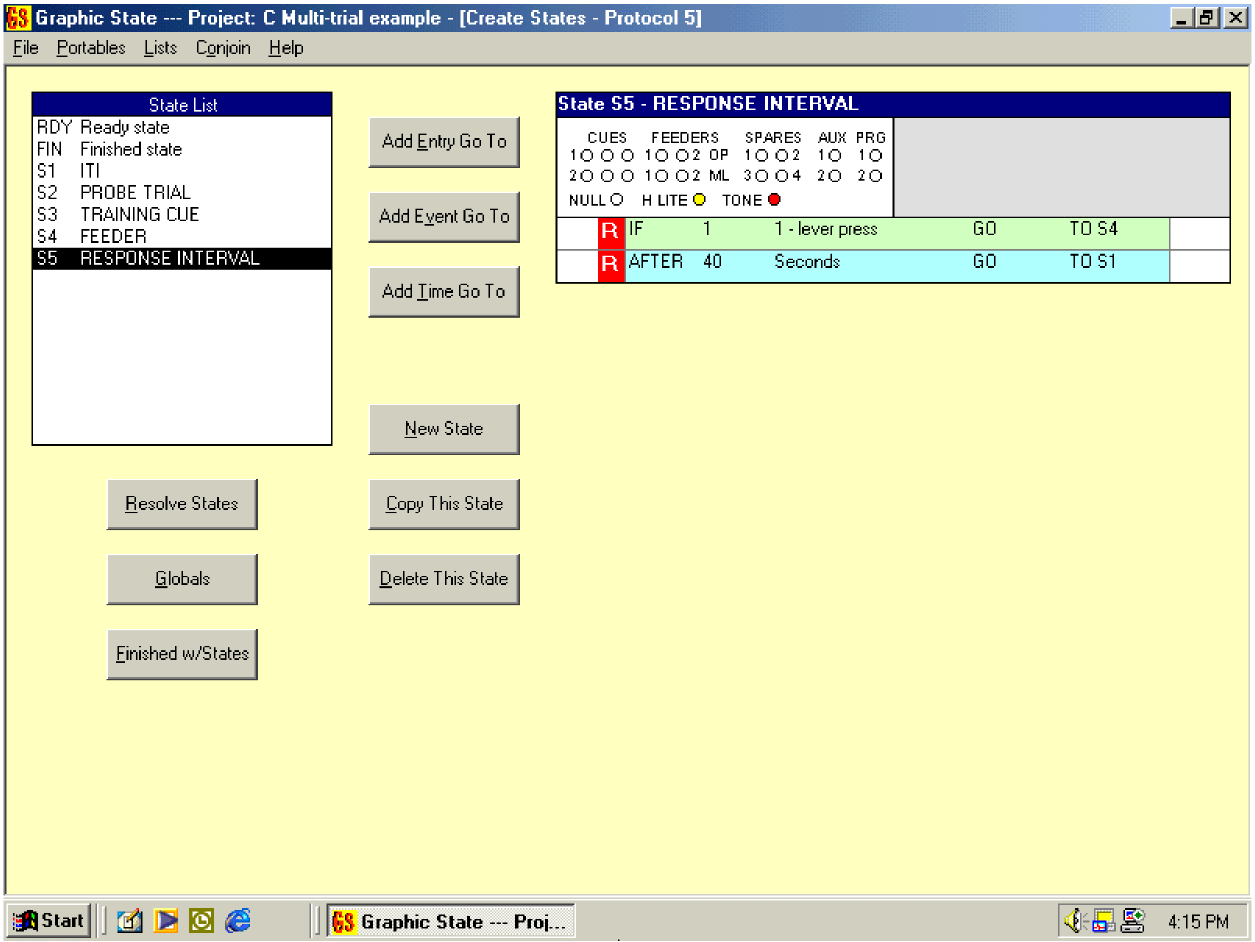
State 2 represents an entire probe trial. The tone is turned on and after 60 seconds control returns to State 1 for the next inter-trial interval.



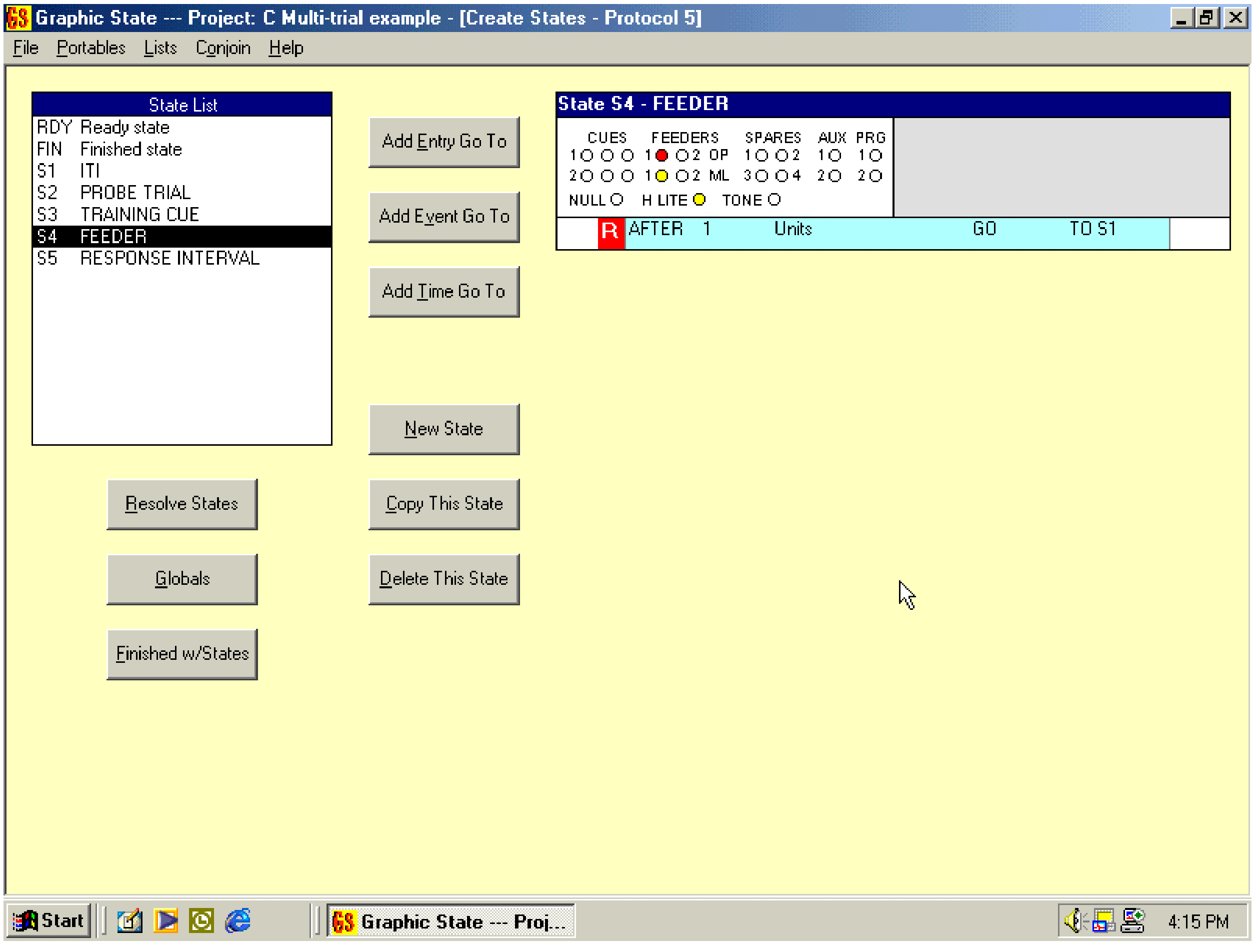
State 3 is the start of a training trial. The tone is turned on and after 20 seconds, control is transferred to State 5.



In State 5, if a lever press response is detected, control is passed to State 4. If there are no lever presses, after 40 seconds control will return to State 1 for the next inter-trial interval.

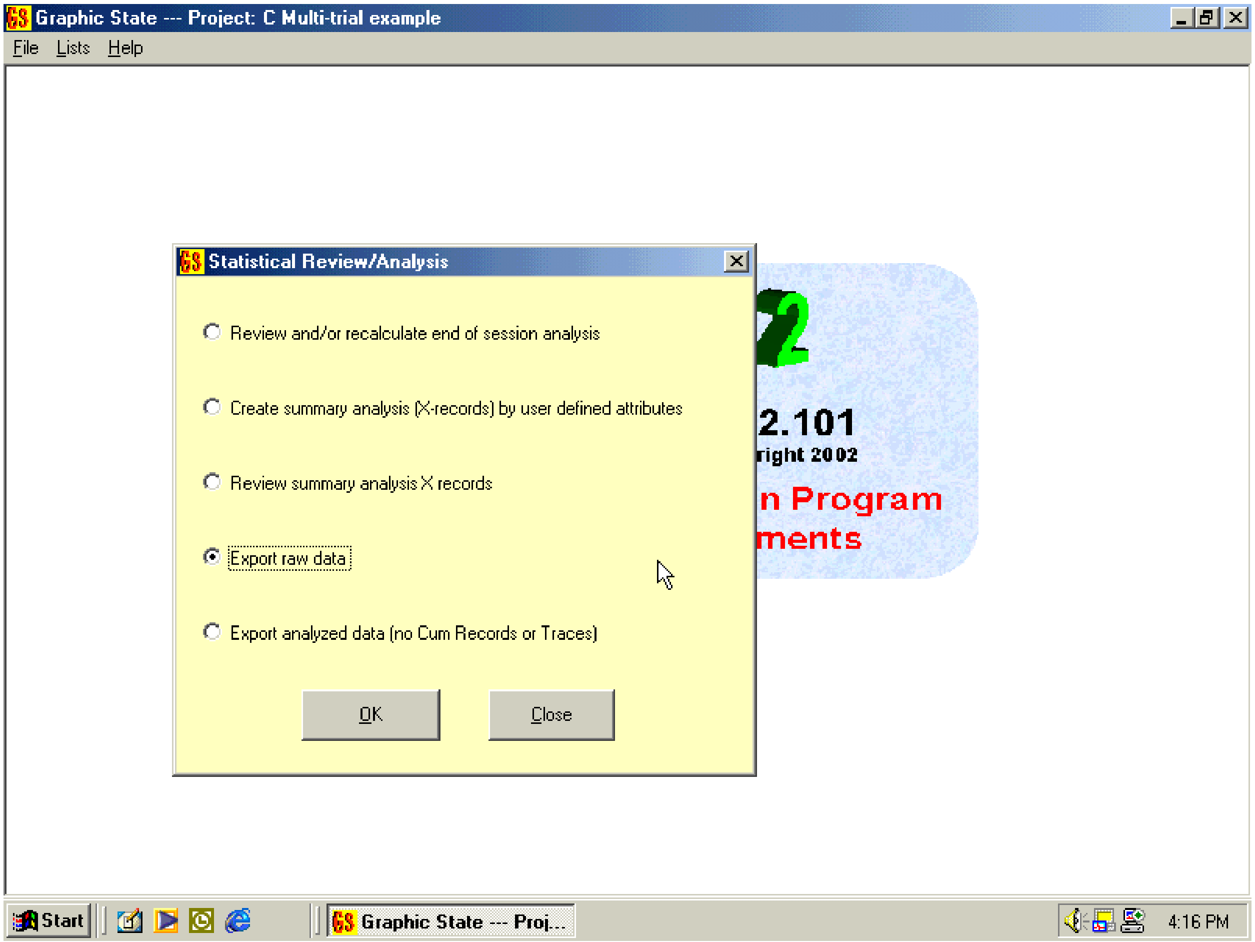


In State 4, one pellet will be delivered and then the trial will immediately end with control returning to State 1.

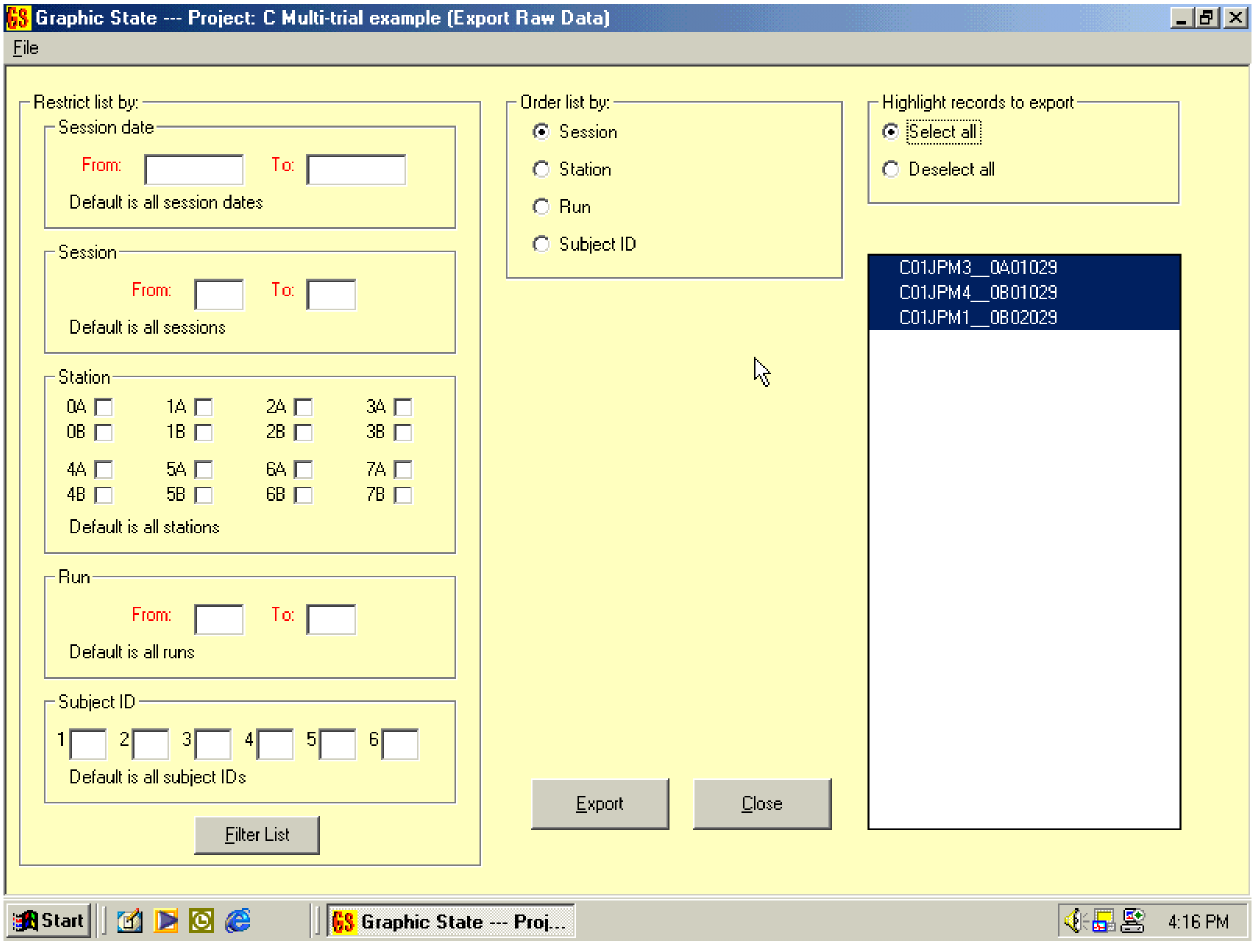


With the Analysis Elements provided with Graphic State Notation, we could create a report of the *average* number of lever presses made during each of the two trial types or the *average* latency to the first lever press during probe trials, etc. But there is no prewritten element that will report these things on a *trial-by-trial* basis. This is relatively easy to do, however, by post-processing exported raw data.

To export raw data, begin by selecting Statistical Review/Analysis from the File menu, and then select the “Export raw data” option and click “OK”.



This will bring up a "filter" screen that allows you to select the run or runs to be included in the exported data. I have selected all of the runs (there are three in my sample database).



After selecting the appropriate runs, click on the Export button and supply a file name and location for the exported raw data file. By default, the file will be placed in the Graphic State Datafiles directory which is in a directory named “2.101-00” (or the version number of your software release) in a directory named “Graphic State.” The structure looks like:

C (or your local hard disk drive)

Graphic State

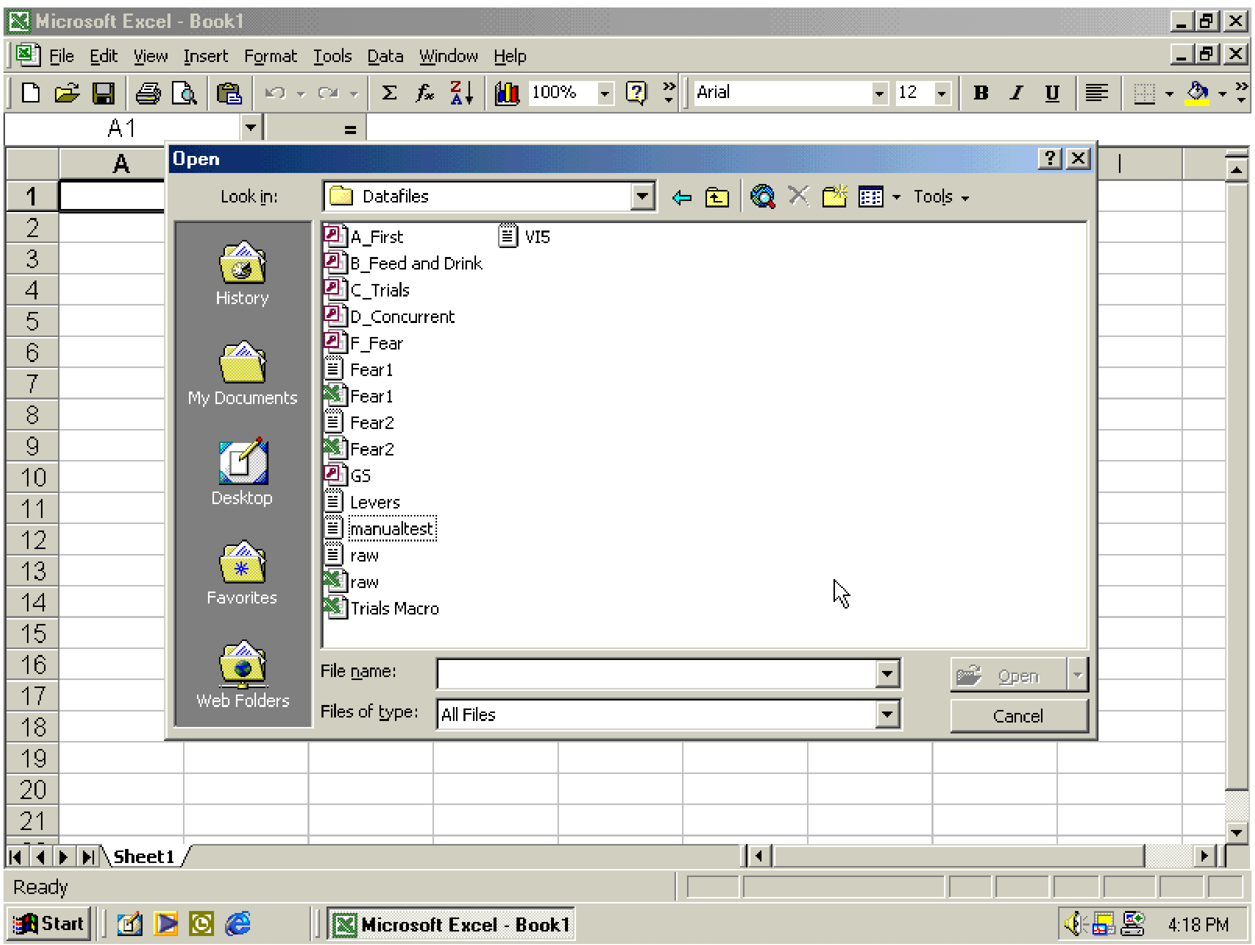
2.101-00

Datafiles

All the Graphic State project databases are also here.

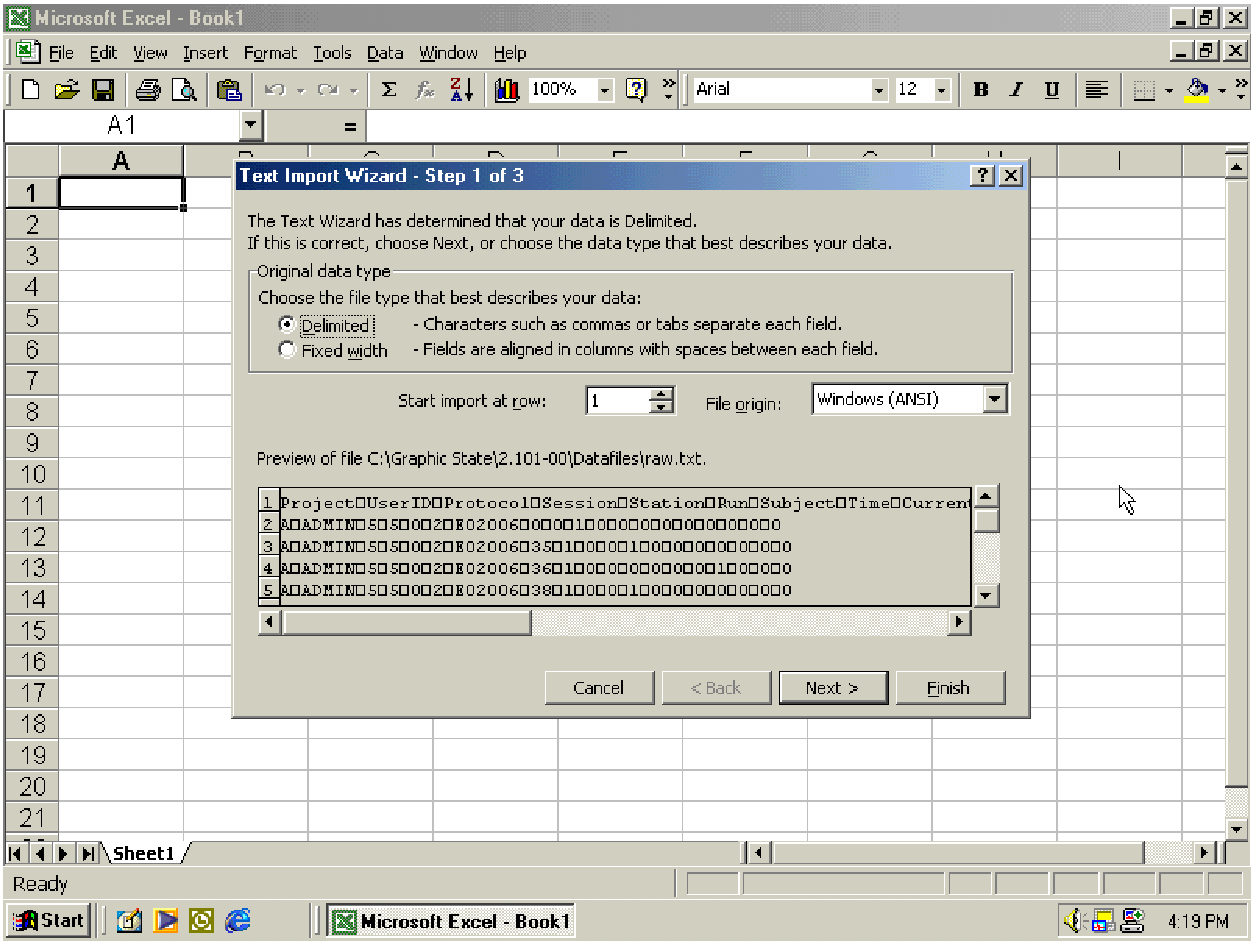
After exiting from Graphic State, you can view and process the exported data using Excel (or another equivalent program).

Start Excel and then select File and then Open. Browse to the directory in which your raw data resides and set the “Files of type:” field to “All files”. You will see something like the screen below. (If you forget to change the file type selection, you will see ONLY any Excel files you already have in this directory.)

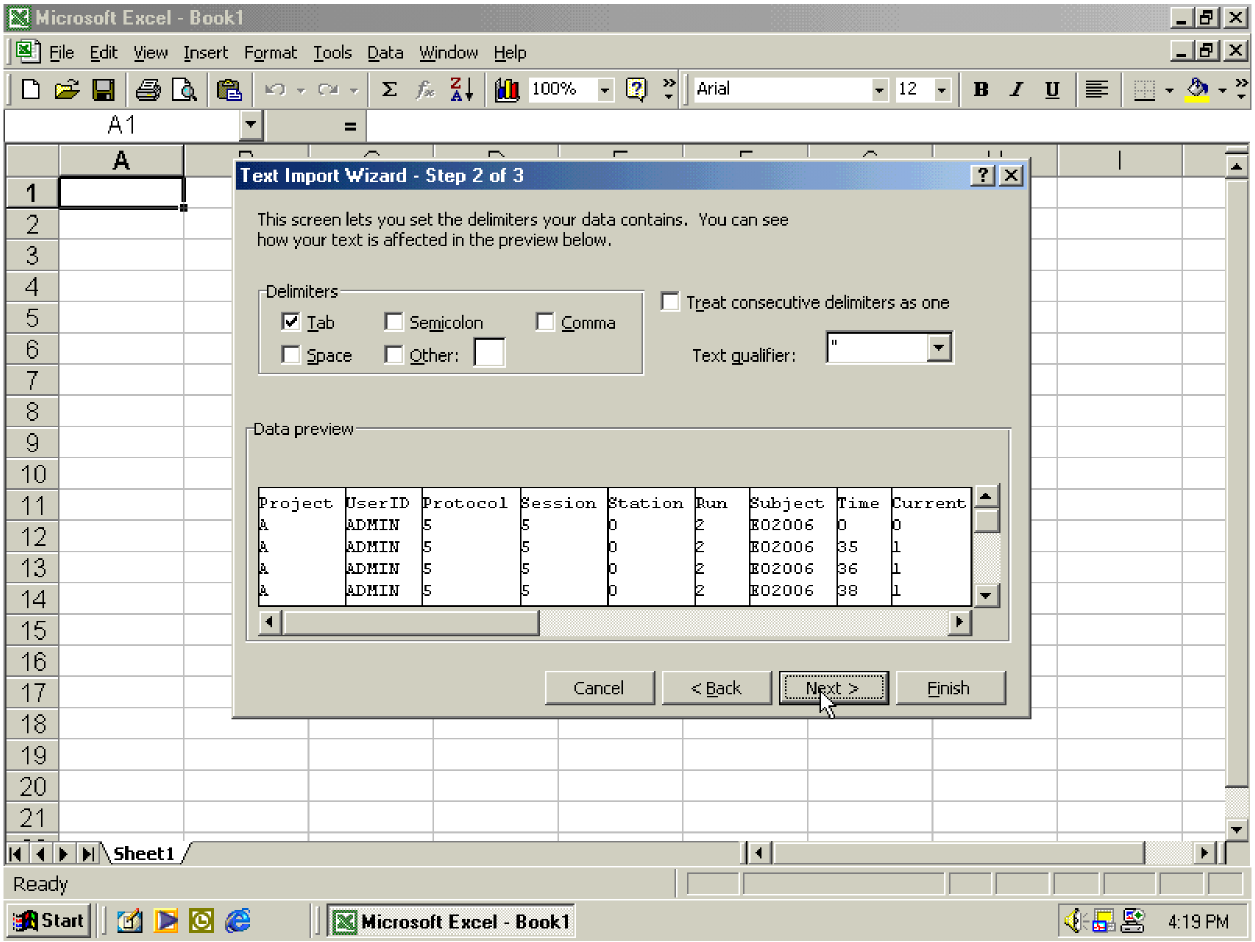


The raw data file you are looking for is a text file (the file extension is “.txt”). The icon looks like a small notepad. The one I created is named “manualtest.”

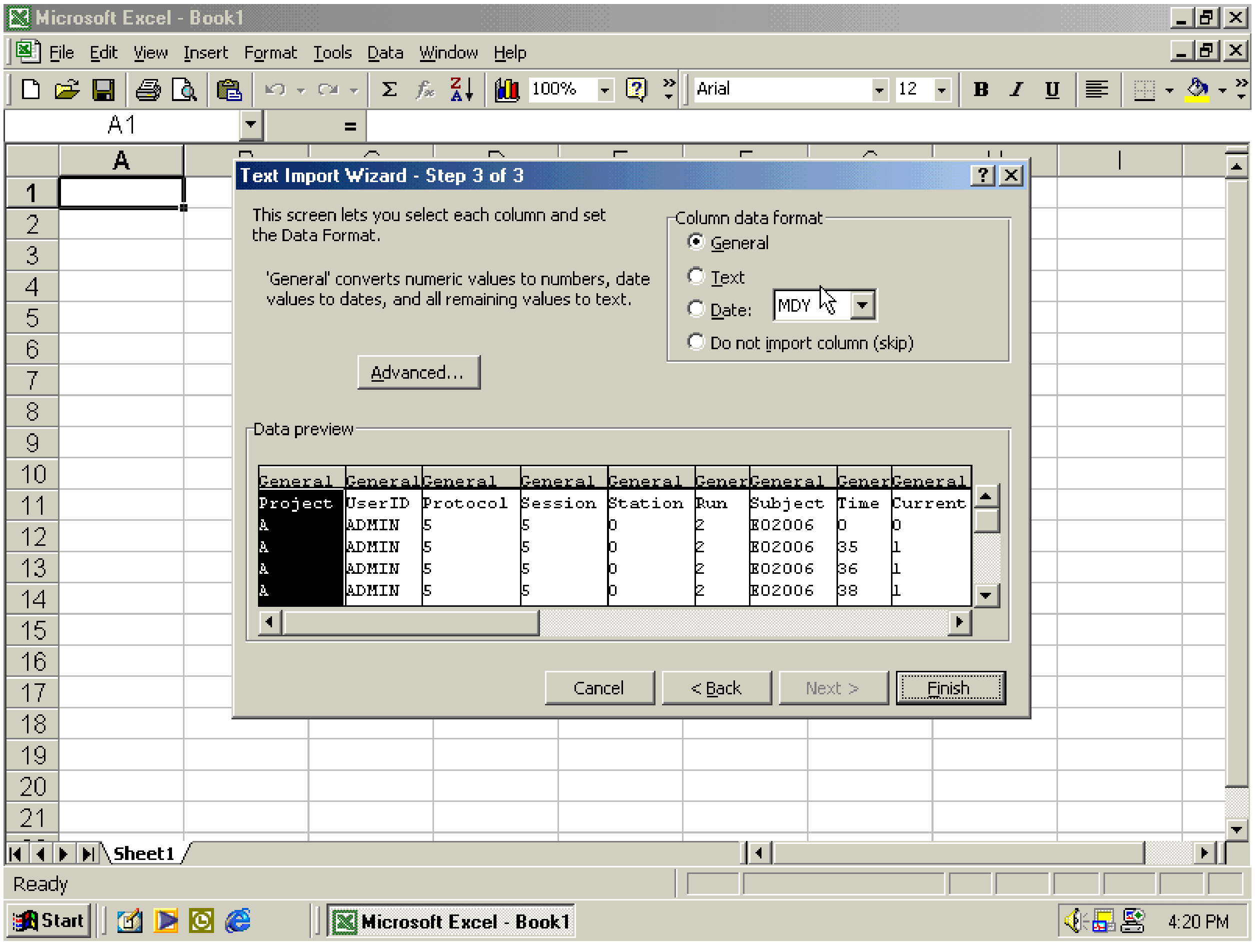
Select the appropriate file and then click on the Open button. The Text Import Wizard will take over.



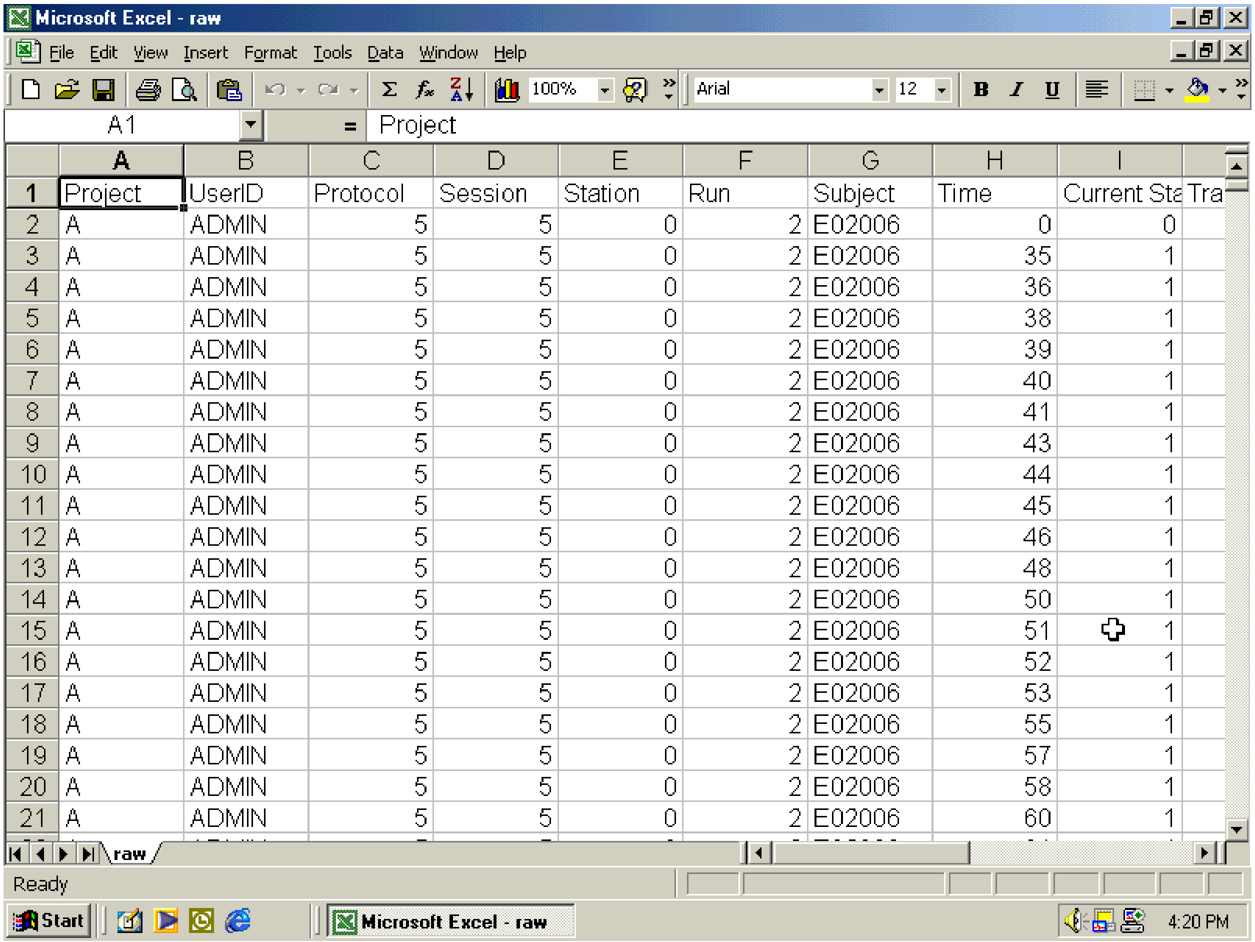
Click on the Next button to accept this default option.



Again, click on the Next button to accept this option.

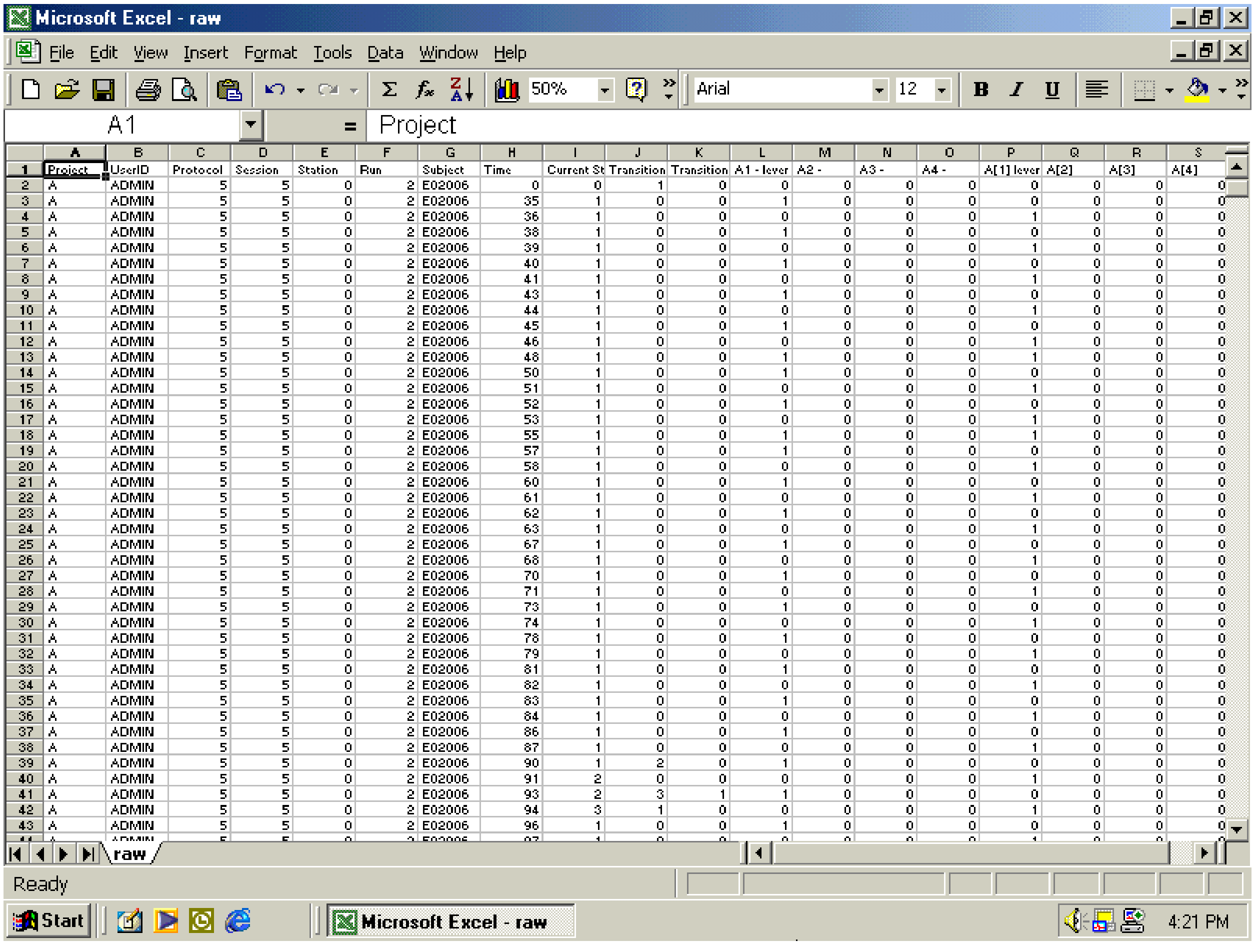


Click on the Finish button to accept the final option. A formatted Excel spreadsheet will appear.



	A	B	C	D	E	F	G	H	I
1	Project	UserID	Protocol	Session	Station	Run	Subject	Time	Current Sta
2	A	ADMIN	5	5	0	2	E02006	0	0
3	A	ADMIN	5	5	0	2	E02006	35	1
4	A	ADMIN	5	5	0	2	E02006	36	1
5	A	ADMIN	5	5	0	2	E02006	38	1
6	A	ADMIN	5	5	0	2	E02006	39	1
7	A	ADMIN	5	5	0	2	E02006	40	1
8	A	ADMIN	5	5	0	2	E02006	41	1
9	A	ADMIN	5	5	0	2	E02006	43	1
10	A	ADMIN	5	5	0	2	E02006	44	1
11	A	ADMIN	5	5	0	2	E02006	45	1
12	A	ADMIN	5	5	0	2	E02006	46	1
13	A	ADMIN	5	5	0	2	E02006	48	1
14	A	ADMIN	5	5	0	2	E02006	50	1
15	A	ADMIN	5	5	0	2	E02006	51	1
16	A	ADMIN	5	5	0	2	E02006	52	1
17	A	ADMIN	5	5	0	2	E02006	53	1
18	A	ADMIN	5	5	0	2	E02006	55	1
19	A	ADMIN	5	5	0	2	E02006	57	1
20	A	ADMIN	5	5	0	2	E02006	58	1
21	A	ADMIN	5	5	0	2	E02006	60	1

By shrinking the “zoom” value on the Excel sheet, you can see more columns without scrolling.



	Project	UserID	Protocol	Session	Station	Run	Subject	Time	Current St	Transition	Transition	A1 - lever	A2 -	A3 -	A4 -	A[1] lever	A[2]	A[3]	A[4]
2	A	ADMIN	5	5	0	2	E02006	0	0	1	0	0	0	0	0	0	0	0	0
3	A	ADMIN	5	5	0	2	E02006	35	1	0	0	1	0	0	0	0	0	0	0
4	A	ADMIN	5	5	0	2	E02006	36	1	0	0	0	0	0	0	1	0	0	0
5	A	ADMIN	5	5	0	2	E02006	38	1	0	0	1	0	0	0	0	0	0	0
6	A	ADMIN	5	5	0	2	E02006	39	1	0	0	0	0	0	0	1	0	0	0
7	A	ADMIN	5	5	0	2	E02006	40	1	0	0	1	0	0	0	0	0	0	0
8	A	ADMIN	5	5	0	2	E02006	41	1	0	0	0	0	0	0	1	0	0	0
9	A	ADMIN	5	5	0	2	E02006	43	1	0	0	1	0	0	0	0	0	0	0
10	A	ADMIN	5	5	0	2	E02006	44	1	0	0	0	0	0	0	1	0	0	0
11	A	ADMIN	5	5	0	2	E02006	45	1	0	0	1	0	0	0	0	0	0	0
12	A	ADMIN	5	5	0	2	E02006	46	1	0	0	0	0	0	0	1	0	0	0
13	A	ADMIN	5	5	0	2	E02006	48	1	0	0	1	0	0	0	1	0	0	0
14	A	ADMIN	5	5	0	2	E02006	50	1	0	0	1	0	0	0	0	0	0	0
15	A	ADMIN	5	5	0	2	E02006	51	1	0	0	0	0	0	0	1	0	0	0
16	A	ADMIN	5	5	0	2	E02006	52	1	0	0	1	0	0	0	0	0	0	0
17	A	ADMIN	5	5	0	2	E02006	53	1	0	0	0	0	0	0	1	0	0	0
18	A	ADMIN	5	5	0	2	E02006	55	1	0	0	1	0	0	0	1	0	0	0
19	A	ADMIN	5	5	0	2	E02006	57	1	0	0	1	0	0	0	0	0	0	0
20	A	ADMIN	5	5	0	2	E02006	58	1	0	0	0	0	0	0	1	0	0	0
21	A	ADMIN	5	5	0	2	E02006	60	1	0	0	1	0	0	0	0	0	0	0
22	A	ADMIN	5	5	0	2	E02006	61	1	0	0	0	0	0	0	1	0	0	0
23	A	ADMIN	5	5	0	2	E02006	62	1	0	0	1	0	0	0	0	0	0	0
24	A	ADMIN	5	5	0	2	E02006	63	1	0	0	0	0	0	0	1	0	0	0
25	A	ADMIN	5	5	0	2	E02006	67	1	0	0	1	0	0	0	0	0	0	0
26	A	ADMIN	5	5	0	2	E02006	68	1	0	0	0	0	0	0	1	0	0	0
27	A	ADMIN	5	5	0	2	E02006	70	1	0	0	1	0	0	0	0	0	0	0
28	A	ADMIN	5	5	0	2	E02006	71	1	0	0	0	0	0	0	1	0	0	0
29	A	ADMIN	5	5	0	2	E02006	73	1	0	0	1	0	0	0	0	0	0	0
30	A	ADMIN	5	5	0	2	E02006	74	1	0	0	0	0	0	0	1	0	0	0
31	A	ADMIN	5	5	0	2	E02006	78	1	0	0	1	0	0	0	0	0	0	0
32	A	ADMIN	5	5	0	2	E02006	79	1	0	0	0	0	0	0	1	0	0	0
33	A	ADMIN	5	5	0	2	E02006	81	1	0	0	1	0	0	0	0	0	0	0
34	A	ADMIN	5	5	0	2	E02006	82	1	0	0	0	0	0	0	1	0	0	0
35	A	ADMIN	5	5	0	2	E02006	83	1	0	0	1	0	0	0	0	0	0	0
36	A	ADMIN	5	5	0	2	E02006	84	1	0	0	0	0	0	0	1	0	0	0
37	A	ADMIN	5	5	0	2	E02006	86	1	0	0	1	0	0	0	0	0	0	0
38	A	ADMIN	5	5	0	2	E02006	87	1	0	0	0	0	0	0	1	0	0	0
39	A	ADMIN	5	5	0	2	E02006	90	1	2	0	1	0	0	0	0	0	0	0
40	A	ADMIN	5	5	0	2	E02006	91	2	0	0	0	0	0	0	1	0	0	0
41	A	ADMIN	5	5	0	2	E02006	93	2	3	1	1	0	0	0	0	0	0	0
42	A	ADMIN	5	5	0	2	E02006	94	3	1	0	0	0	0	0	1	0	0	0
43	A	ADMIN	5	5	0	2	E02006	96	1	0	0	1	0	0	0	0	0	0	0

The first seven columns of the spreadsheet identify the Project letter, User ID, Protocol number, Session number, Station number, Run number and Subject ID. These values will be the same for every row for a particular run. This file may contain more than one run, however, and these identifying columns can be used to tell where the next run begins.

There is a time-stamped row for each sample period during which *either* a switch input was recorded *or* a switch offset was recorded *or* a state transition occurred. If nothing happened during a sample, nothing was recorded. This eliminates unnecessary data and keeps the file size manageable.

Following the identifier columns, the columns and their meanings are:

Time

This contains the elapsed time, in units or number of sample periods, since the start of this run. If the sample size selected was 100 ms, for example, each unit equals 0.1 seconds.

Current State

This is the number of the state currently in control. The RDY state is represented by 0 and the FIN state is represented by -1.

Transition State

If a transition to a new state has been triggered, the field contains the number of the new state. If no transition is occurring, this field is 0.

Transition Event

If a transition has been triggered by an event (i.e. by a switch input), this field is 1. If the transition was triggered by time, or if there is no transition, this field is 0.

Input switch onset fields A1 – A4

The next four fields report onset events from the first four input switches associated with the Linc. If the switches were named in the protocol, the names also appear here. In our example, Switch A1 was named “lever” and column L has a heading “A1 – lever”.

In each of these columns, a 1 indicates that a switch onset was recorded (e.g. when an animal pressed a lever) and a 0 indicates that one was not.

Input switch offset fields {A1] – [A4]

These fields are identical to the last four except that they report when a switch offset occurred (e.g. when the animal released the lever).

Additional switch onset and offset fields

Additional sets of switch onsets and offsets will appear in subsequent columns if they were part of the station definition for the project (if you used a full Linc instead of a half Linc or even multiple Lincs per station).

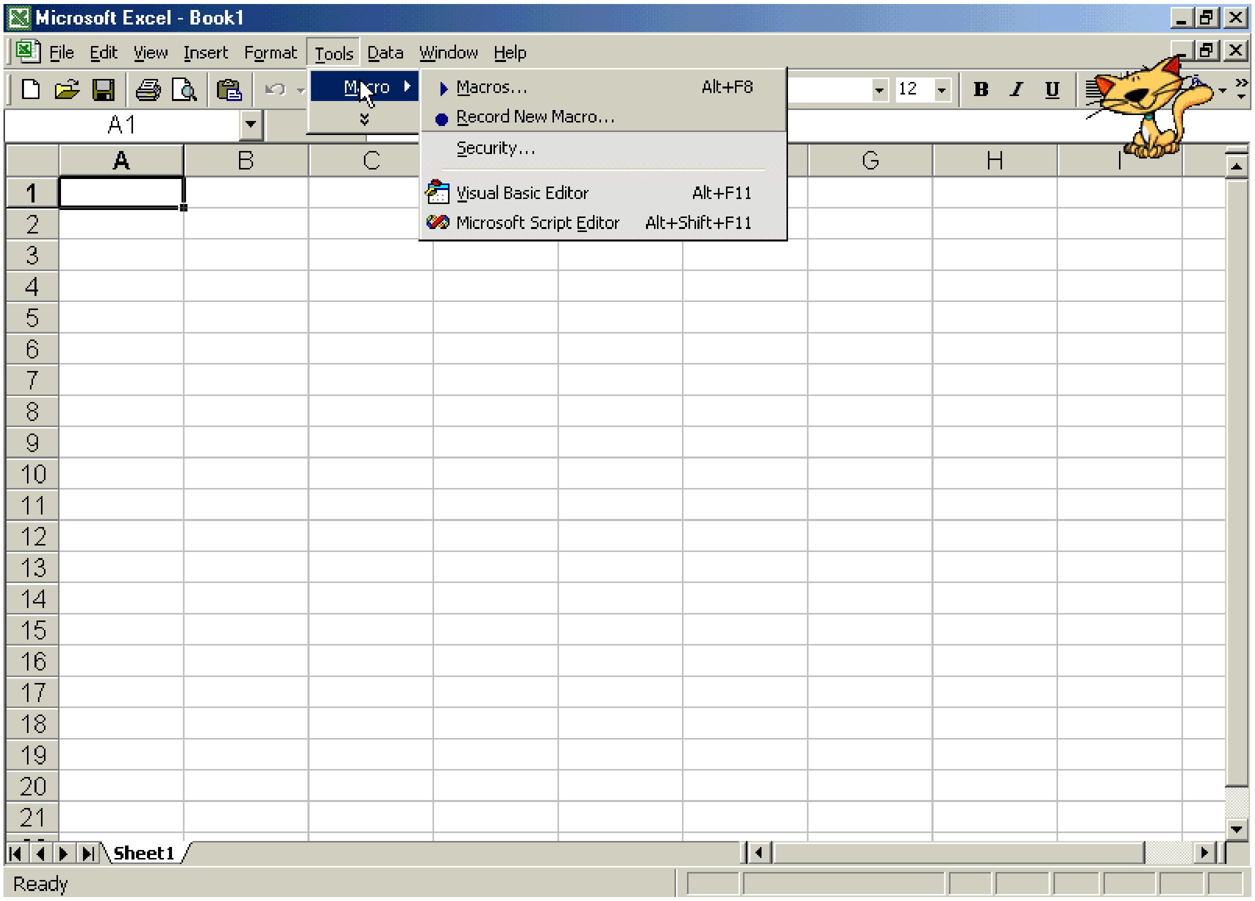
To save the spreadsheet for later processing, select File and then Save As. Specify a file name and select “Microsoft Excel Workbook” in the “Save as type” field.

You may, of course, use any Excel processing that you find appropriate for analyzing the raw data. The rest of this tutorial chapter will describe how to create an example Excel macro (a program in Visual Basic that runs from within Excel) to produce a report from this particular sample raw data.

To refresh your memory, this protocol runs 60 trials in total, 30% of one type (“training”) and 70% of a second type (“probe”), in random order. The trials are separated by a variable inter-trial interval. On training trials, a tone is sounded for 20 seconds and then, if a lever press occurs anytime during the next 40 seconds, while the tone continues, a reinforcer is delivered and the trial ends. If there is no lever press, the trial ends after the 40 seconds with no reinforcer delivery. On probe trials, the tone sounds for the full 60 seconds with no reinforcer deliveries.

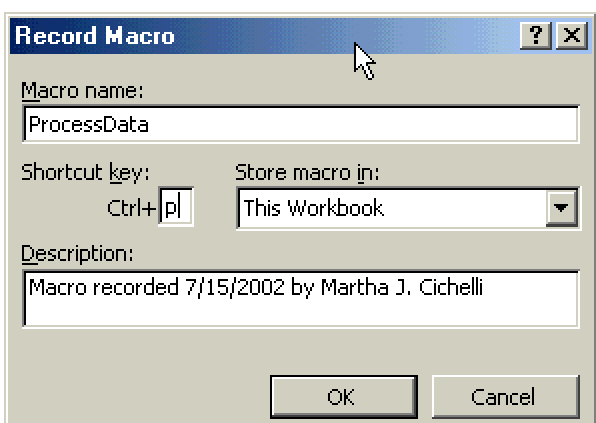
What we want to do is to report each trial separately, listing the length of the preceding inter-trial interval, the type of trial (training or probe), the latency to the first lever press on that trial, the total length of the trial, the total number of lever presses (switch 1), and the total number of responses on switch 2.

The easiest way to begin to create an Excel macro is to record a sample macro and then to edit it. Open a new Excel spreadsheet. From Tools on the toolbar, select Macro and then Record New Macro.



This will bring up a dialog box in which you can name your sample macro and specify a shortcut key sequence for invoking it. I named mine “ProcessData” and selected Ctrl-p as the shortcut key.

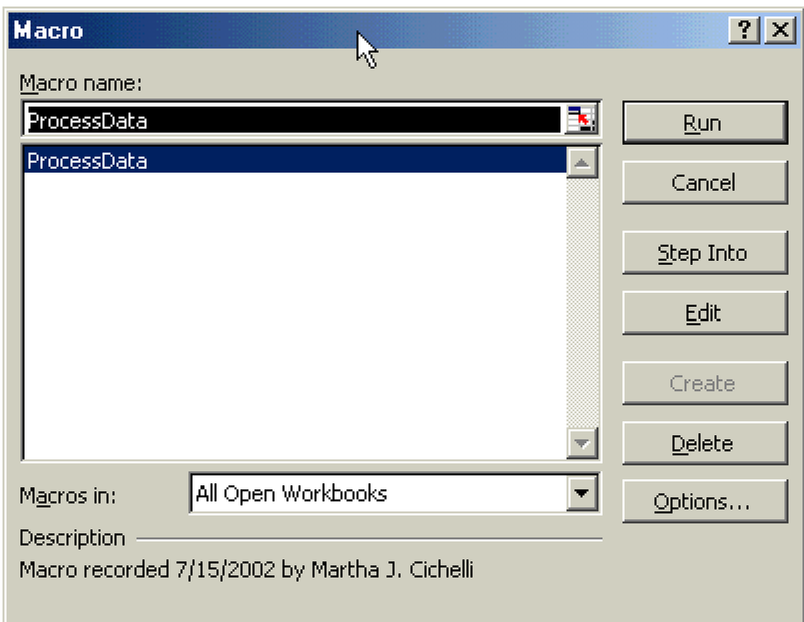
Click OK to continue.



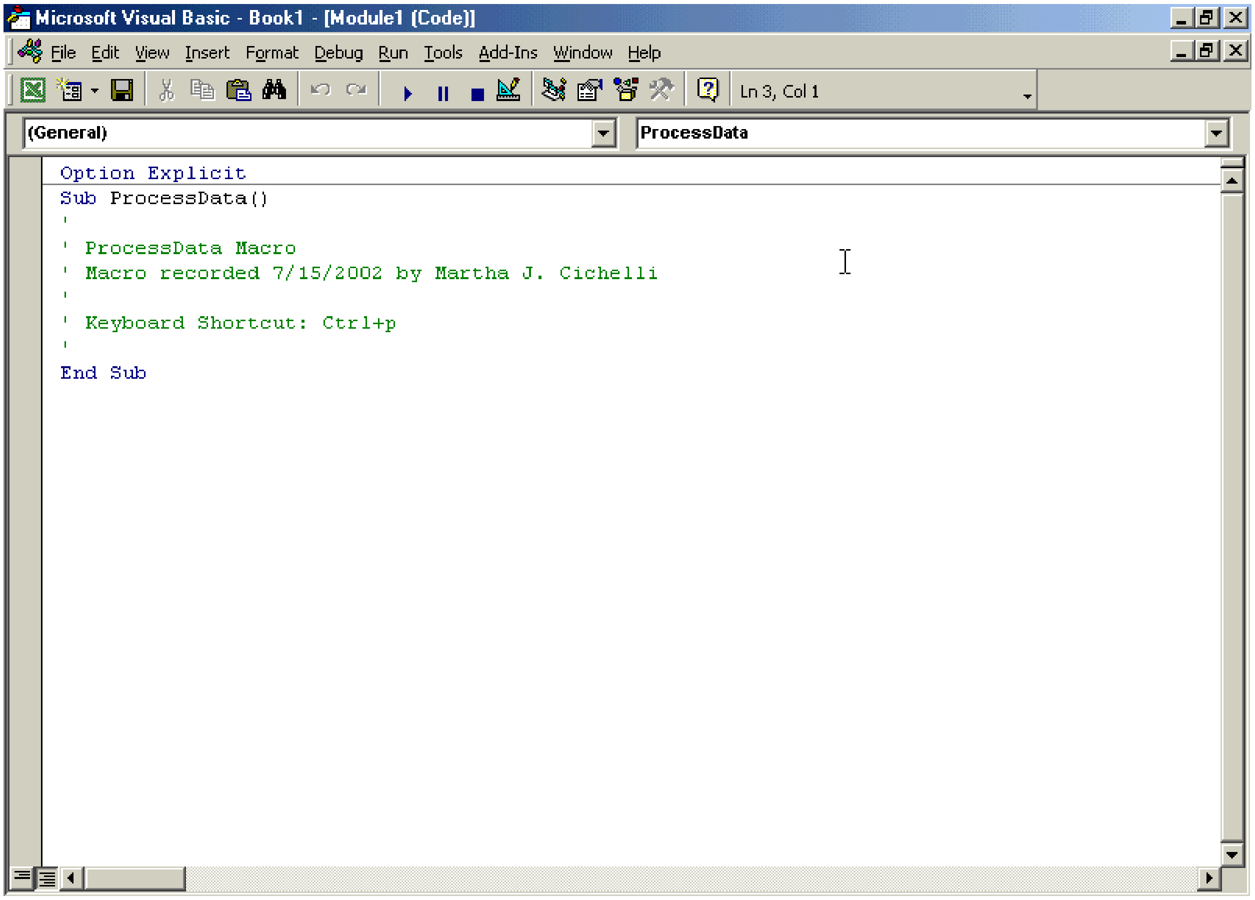
Now, everything you do within Excel, until you stop recording, will become part of the macro called ProcessData. This is a useful way to find out what the “behind the scenes” code looks like for any Excel processing.

We can stop recording immediately just to create the basic macro structure. From Tools on the toolbar, select Macro and then Stop Recording.

To see what we have created, from Tools, select Macro and then Macros. This will produce a list of currently available macros.



Highlight the desired macro, ProcessData, and click on the Edit button. What you will see is the following macro shell.



If you are familiar with Visual Basic, you will know how to expand this to do whatever processing you want. If not, try experimenting first by recording while you do additional Excel processing steps "by hand". There are also a number of good introductory Excel macro processing books (e.g. in the "… for Dummies" series).

I have listed, in an appendix, the macro I wrote to process this sample data. You could always copy this code and modify it as required for your needs.

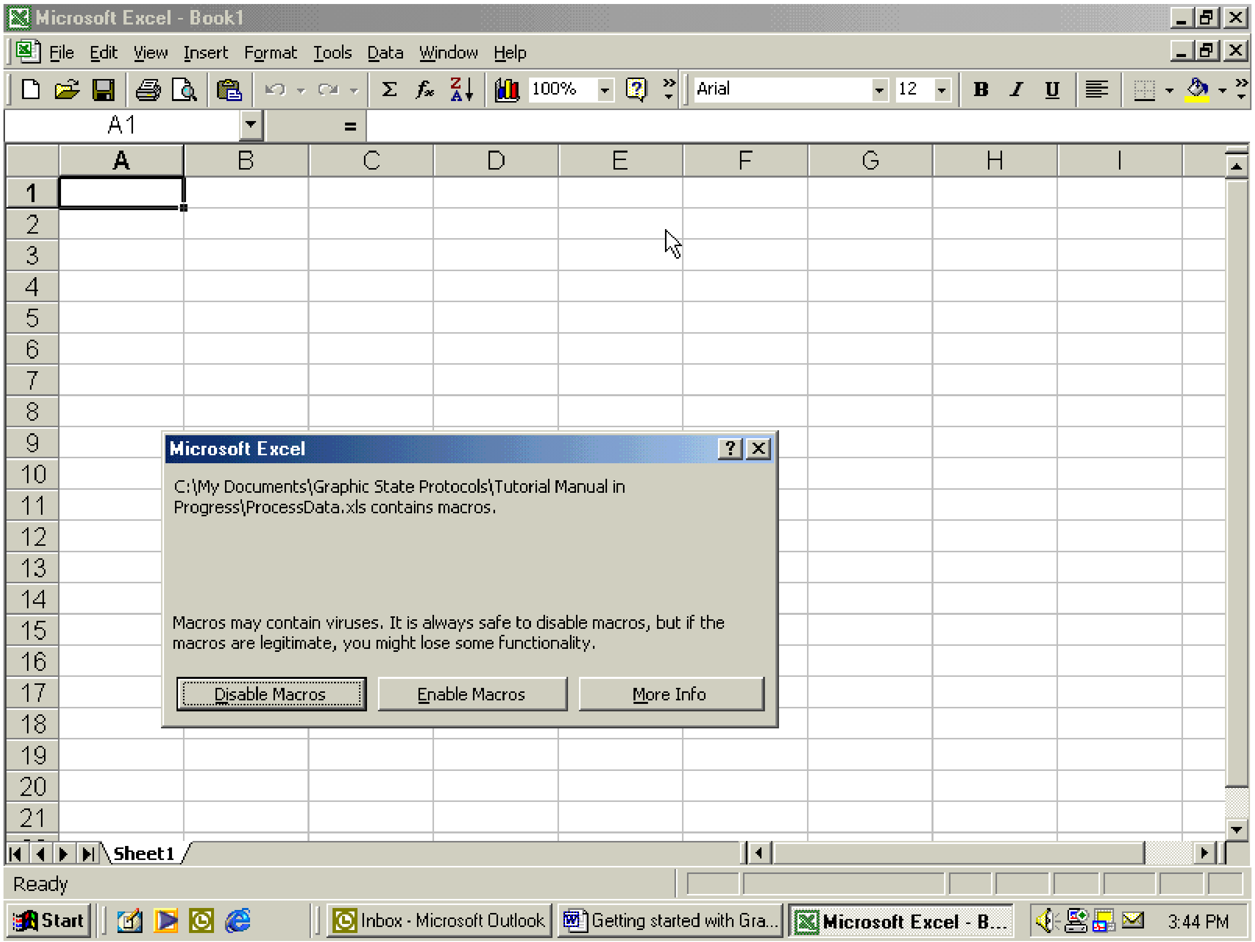
As you create your macro code, save it (in its own worksheet) frequently. The code syntax is checked interactively as you type. Errors are noted in red immediately. Logic errors that are not syntax errors can only be detected by testing. It is a good idea to write your code in small increments and test after each such increment.

To test (or run) your macro, open another worksheet containing your input raw data. You will have *both* the spreadsheet containing your macro and another one containing your raw data open at the same time. Position the cursor anywhere in the raw data and type the hotkey (ctrl-p in our case). This will run the macro. If it runs to the end, you can then look in the data spreadsheet to see your output. If it causes an error partway through, you will automatically be placed into a debugger which highlights the location of the error.

You can learn how to use the debugger by experimentation and extensive use of the help facility, or you can refer to a good tutorial book.

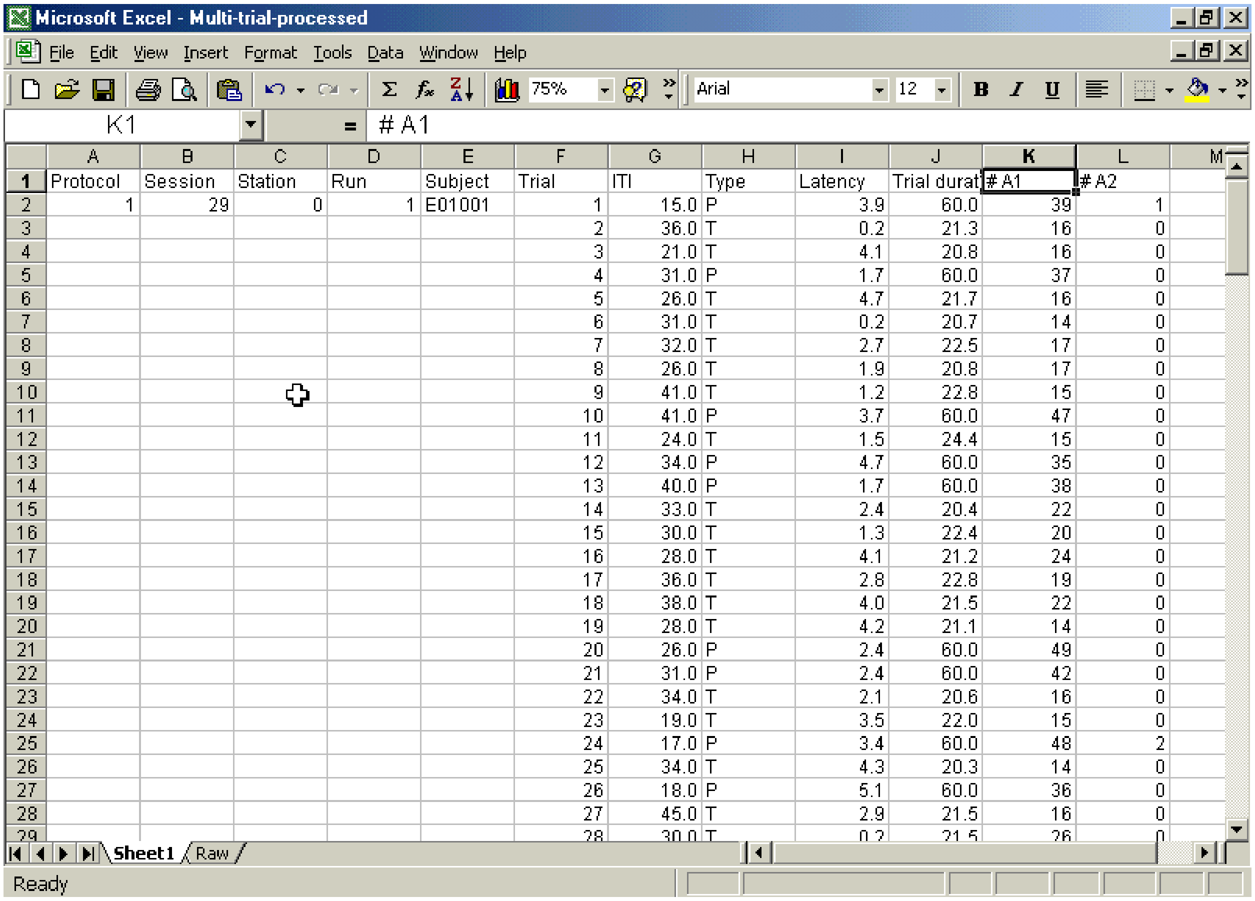
Once the macro is completed and working correctly, it can be run on any exported data file (for this protocol). Start Excel and open the spreadsheet that contains the macro. I called my macro spreadsheet by the same name as the macro (ProcessData) just for convenience.

When you open any spreadsheet that contains macros, you will see the following dialog. Macros from unknown sources may contain viruses and this message alerts you to the presence of macros in this file. We *want* this macro, however, so click on the Enable Macros button.



Without closing this spreadsheet, open your raw data spreadsheet (either the text file as it was exported from Graphic State Notation or a saved version after processing it with the Import Wizard). With the cursor anywhere in the raw data, type the hotkey (ctrl-p) to invoke the macro.

What follows is a snapshot of part of the processed data spreadsheet for my sample data.



	A	B	C	D	E	F	G	H	I	J	K	L
1	Protocol	Session	Station	Run	Subject	Trial	ITI	Type	Latency	Trial durat	# A1	# A2
2	1	29	0	1	E01001	1	15.0	P	3.9	60.0	39	1
3						2	36.0	T	0.2	21.3	16	0
4						3	21.0	T	4.1	20.8	16	0
5						4	31.0	P	1.7	60.0	37	0
6						5	26.0	T	4.7	21.7	16	0
7						6	31.0	T	0.2	20.7	14	0
8						7	32.0	T	2.7	22.5	17	0
9						8	26.0	T	1.9	20.8	17	0
10						9	41.0	T	1.2	22.8	15	0
11						10	41.0	P	3.7	60.0	47	0
12						11	24.0	T	1.5	24.4	15	0
13						12	34.0	P	4.7	60.0	35	0
14						13	40.0	P	1.7	60.0	38	0
15						14	33.0	T	2.4	20.4	22	0
16						15	30.0	T	1.3	22.4	20	0
17						16	28.0	T	4.1	21.2	24	0
18						17	36.0	T	2.8	22.8	19	0
19						18	38.0	T	4.0	21.5	22	0
20						19	28.0	T	4.2	21.1	14	0
21						20	26.0	P	2.4	60.0	49	0
22						21	31.0	P	2.4	60.0	42	0
23						22	34.0	T	2.1	20.6	16	0
24						23	19.0	T	3.5	22.0	15	0
25						24	17.0	P	3.4	60.0	48	2
26						25	34.0	T	4.3	20.3	14	0
27						26	18.0	P	5.1	60.0	36	0
28						27	45.0	T	2.9	21.5	16	0
29						28	30.0	T	0.2	21.5	26	0

Once the macro has run, use Save to store the raw and processed data wherever you want to keep it. Develop a naming convention for these files that will help you find them later.

Copying, exporting or importing protocols

You may want to make a copy of a protocol (perhaps to make another with a slight variation), or you may want to send a protocol to another user or receive one from another user. To do any of these, from the File drop-down menu, click on the first option Select Database. Highlight the appropriate database and click on the Select button to make it the active database. Next, from the File menu, select Edit/Review Experiment Protocol.

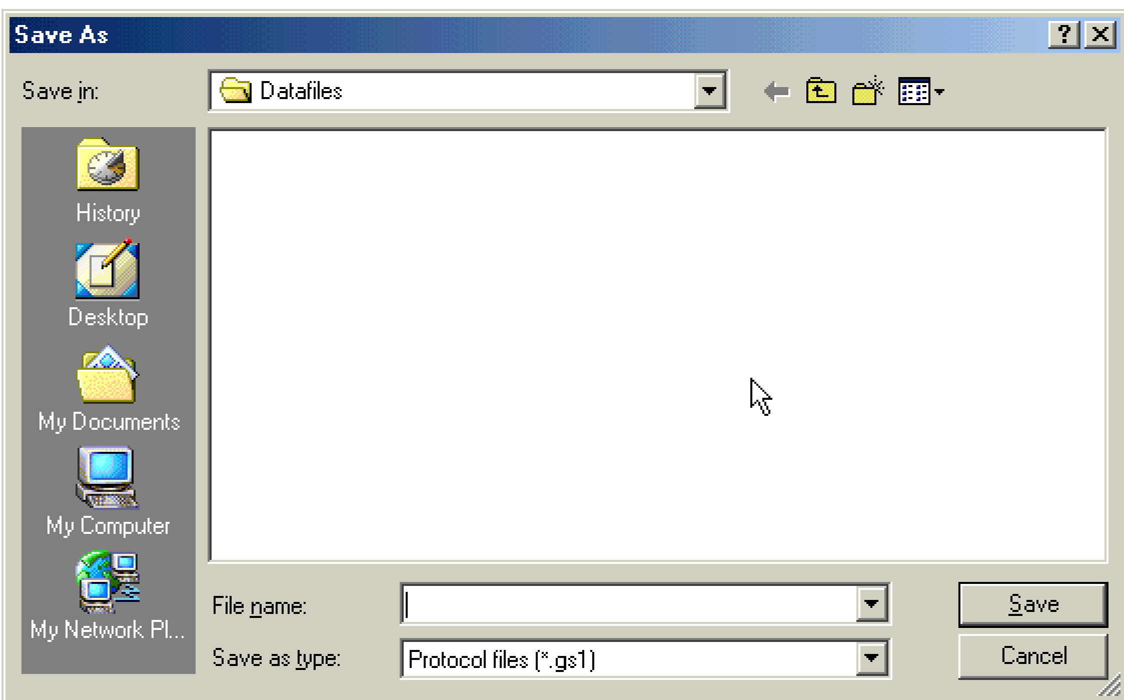
If you are copying a protocol, highlight the appropriate protocol and then click the Copy button. This will create a new protocol, with a new name, that is otherwise an exact copy of the original one. The default new name will look like, for example

(C1) Copy of protocol 1

You may change any part or all of this name. The information in parentheses may be left as is to serve as a trail of the origin of the protocol (if you wish) while the rest of the name could be changed to reflect the new purpose of the protocol.

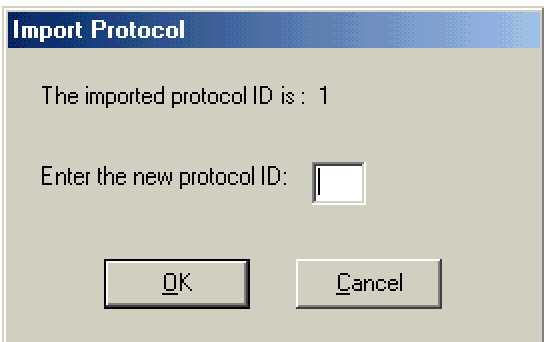
You may now change any part of the protocol. You must, at least, use the Resolve States button on the State Creation screen.

If you want to send the protocol to another user, after selecting it on the Edit/Review screen, click on the Export button. This will bring up the following screen where you may specify the location to write the file.



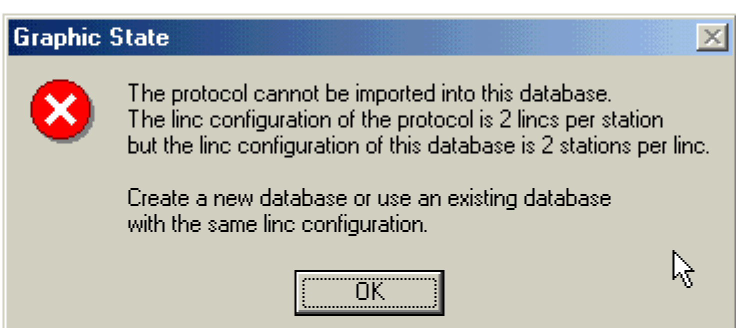
Exported protocols are stored with a suffix of “gs1”. By default, they are stored in the Datafiles directory, but you may instead write the file anywhere on your hard drive or on other media (e.g. a floppy disk). Browse to a new directory, if you wish, and then type in the filename you want to use and click on the Save button.

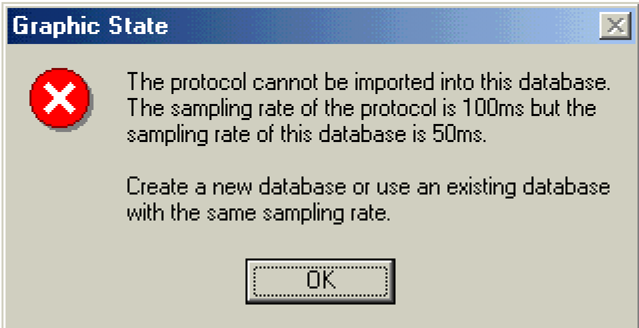
To add a protocol to your database from an outside source, from the Edit/Review screen click on the Import button. This will bring up a very similar screen asking for the location and filename of the protocol to import. This file must be a .gs1 file that was exported from a Graphic State database. Locate the appropriate file and click on the Open button. You will see something like this:



Select the number to be used in your database for the protocol (usually the next available number) and click on OK. If the protocol number that was originally used for the protocol is available in your database, this number will automatically be used instead and you will not see this dialog box.

The protocol to import must match the sample rate and Linc configuration of the current database. If it does not, you will see a message like one of the two below:





In either of these cases, you must create or use a database with the same Linc configuration and sample rate of the protocol you wish to import.

When the protocol is successfully imported, you will then see it in your list of protocols. You may, of course modify it in any way. You will, at least, need to Resolve States.

Keeping your databases safely backed up

The protocols you create and the data you collect are valuable resources that must be protected. Computer equipment is fallible! Never depend **solely** on the working copies of your databases.

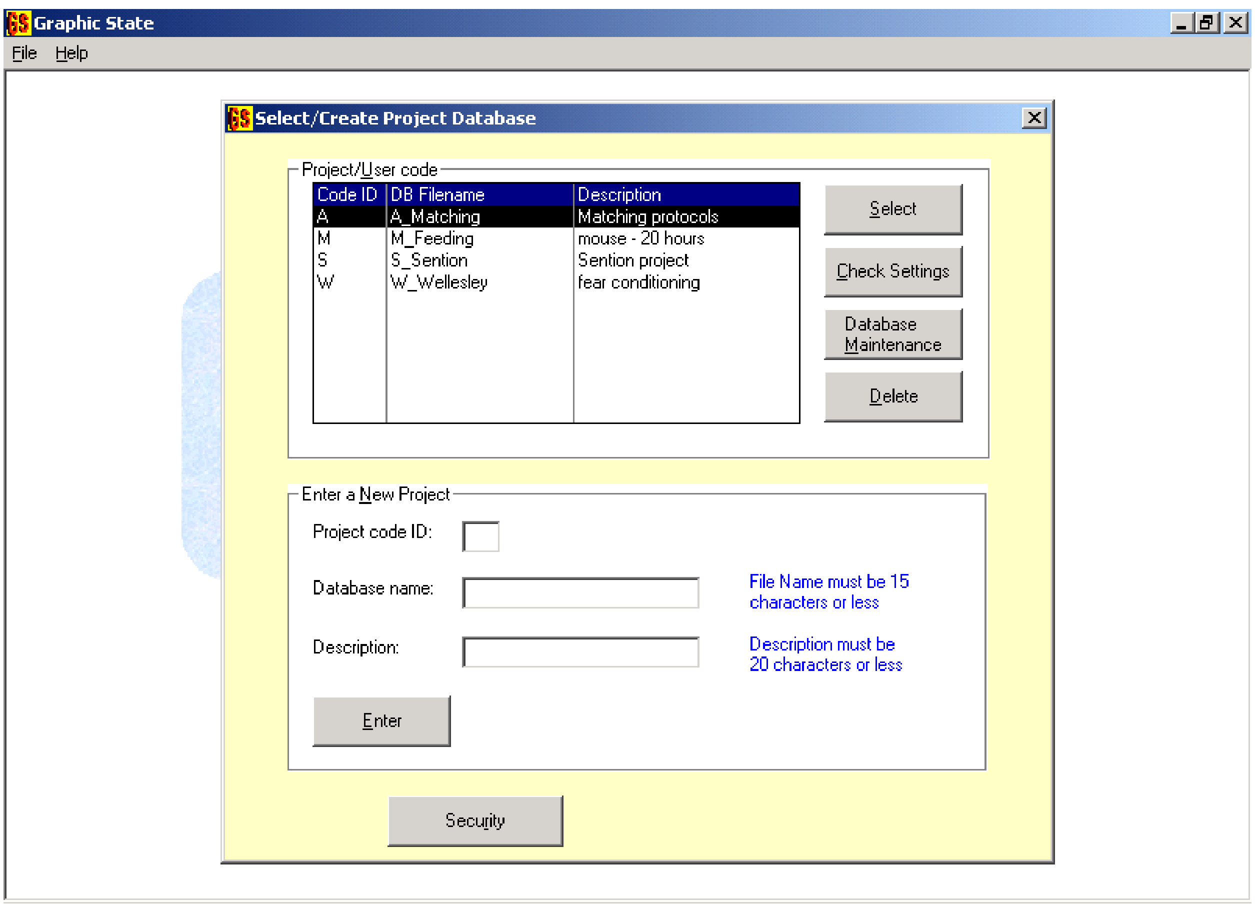
You should make frequent backup copies of your active databases – preferably to external media or to another computer. You can either do this outside of the Graphic State software using Windows copy facilities, or you can do it from within the Graphic State software.

The project databases are Access 97 files. All of the protocol descriptions and all of the collected data are kept in a single database file for each project. You will find these files in the Datafiles directory under the directory for the current version of Graphic State on your primary hard drive. For example:

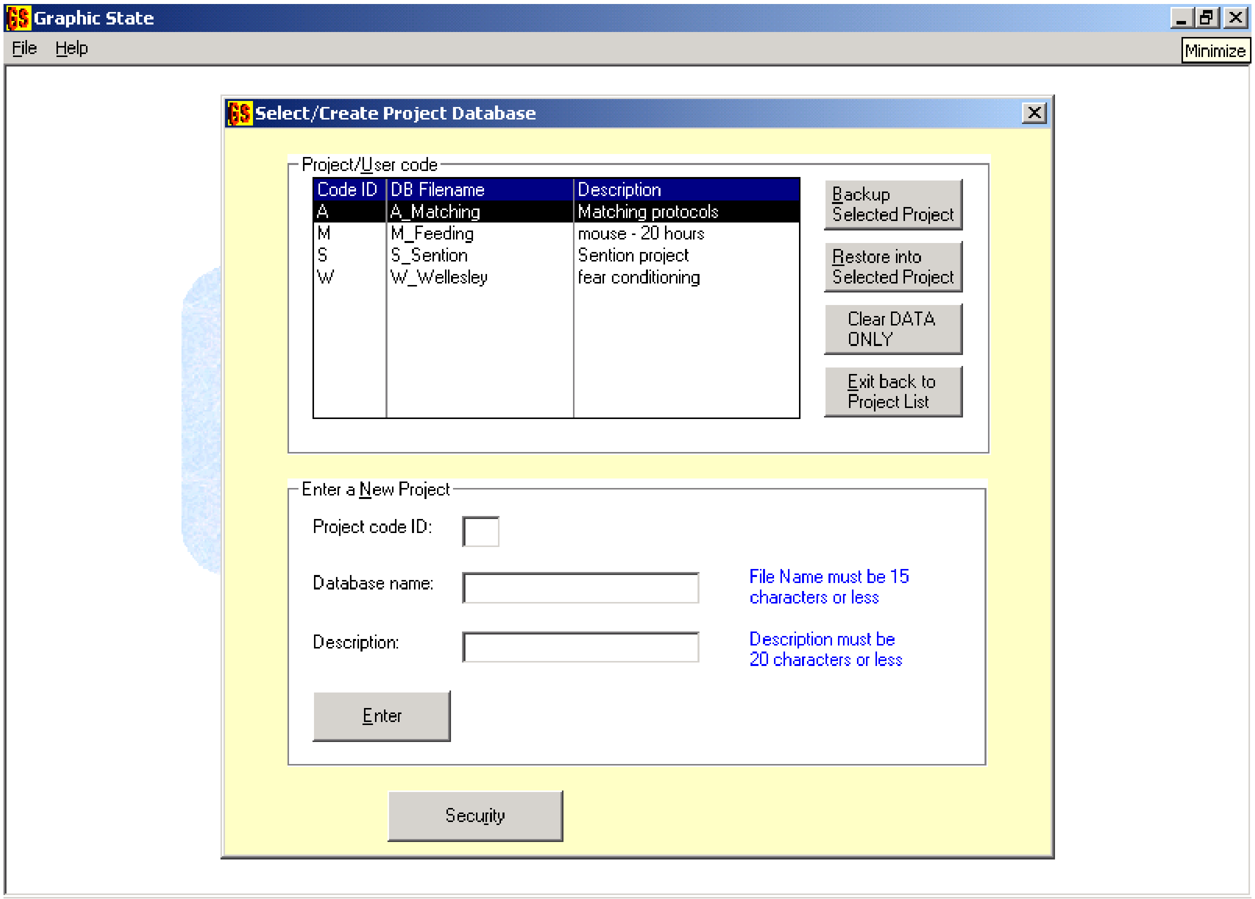
```
C:
    Graphic State
        2.102-00
            Datafiles
```

Each of the projects you have defined (e.g. Project A with name MyProject) will be found in this directory (with the name A_MyProject.mdb). You may use the Windows file copy facility to copy each of these databases to a safe backup location. There are other .mdb files in this directory and in the one above it that are used by Graphic State, but that are not specific to a particular project.

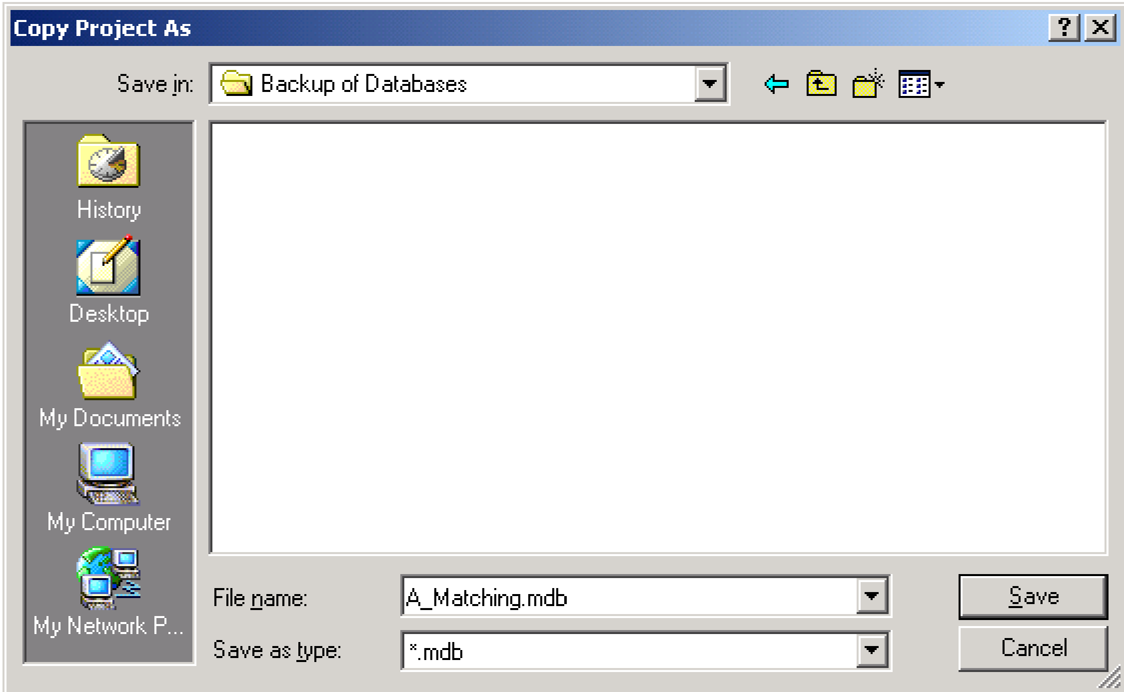
Starting with V2.102-00, you may also make backup copies of databases (and restore them also) from within the Graphic State program. On the Select/Create Project Database screen, you will see a Database Maintenance button.



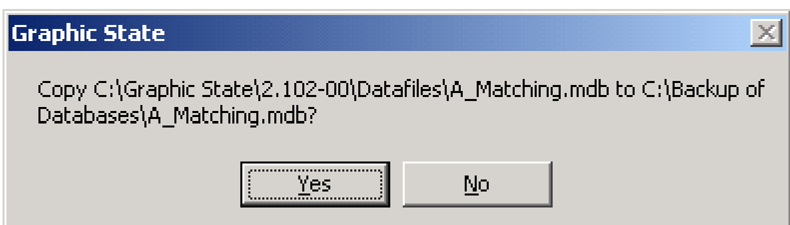
Click on this button to see the maintenance options.



Move the cursor to the project you want to backup. If you select the Backup Selected Project button, you will be given the chance to specify the exact filename and location for the backup copy of the database. Do NOT put your backup copy in the Datafiles directory – pick a new, safe location.

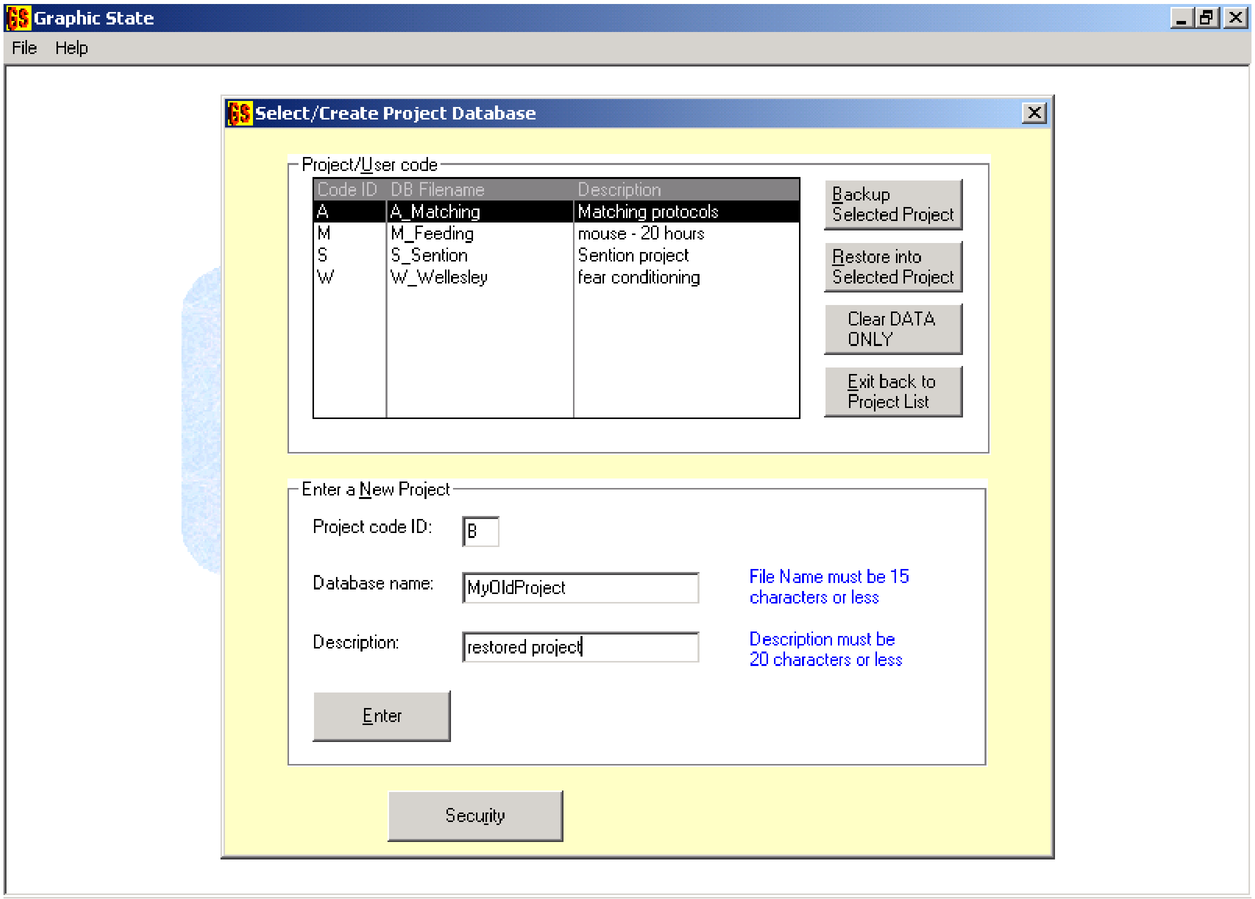


Click on Save and you will get one more chance to confirm your choice.

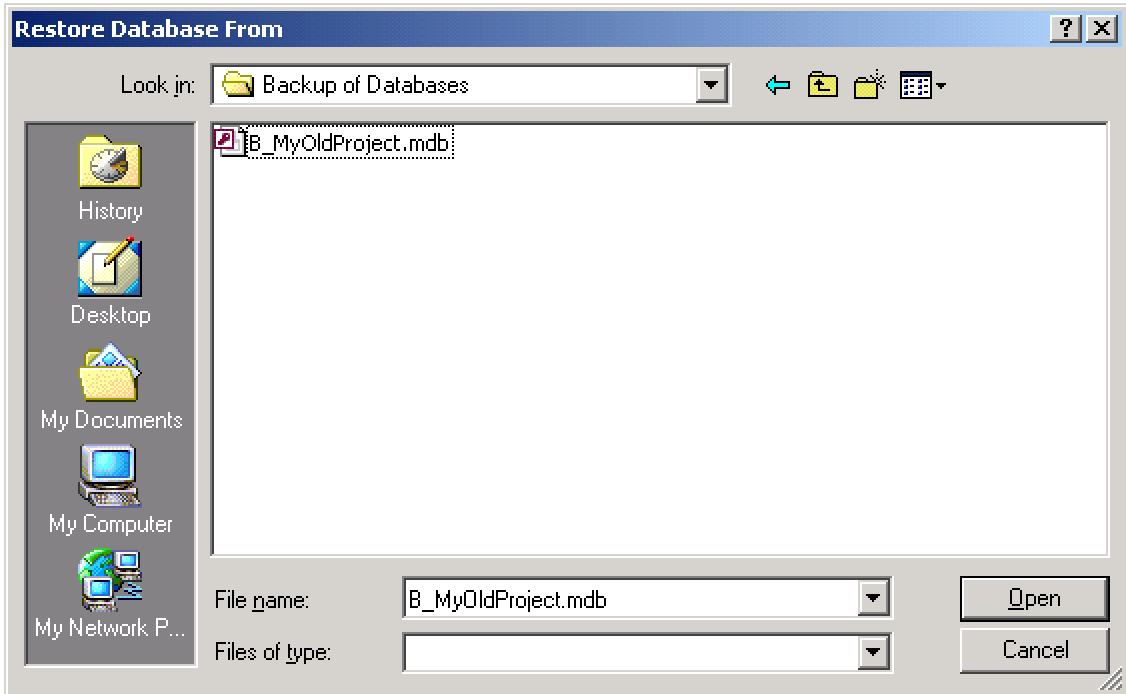


If you need to restore a backup copy of a database into the Datafiles directory, you will use the “Restore into Selected Project” button. Before you do this, make sure you have the project defined in the Graphic State project list. If you want to completely replace an existing project, you will already see it in the project list.

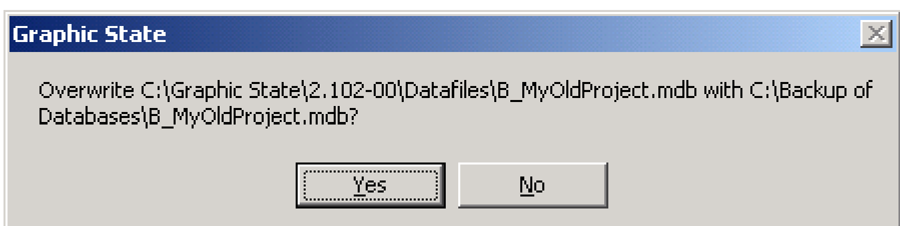
If you are bringing back a database that you have not used in some time and that you have already deleted from the project list, first use the lower part of the screen to “Enter a New Project”. It is important to give it the name you want to use and to select the correct sample rate and Linc configuration, but you do NOT need to enter any protocols because you will be covering them up with the database you will restore.



With the correct project selected, click the Restore button. Specify the location of the database to be restored, and click on the Open button.



Again, you will get a final chance to verify your choice.

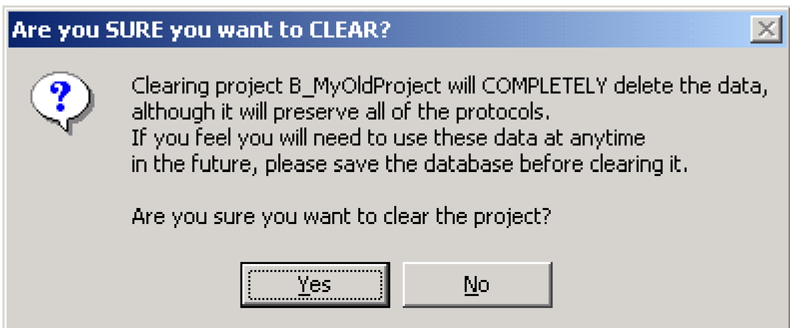


There is another option on the Database Maintenance screen labeled "Clear DATA ONLY". This is used only if you have reached or are approaching the limit of 999 experimental sessions per database. You cannot continue with a database that has reached this limit – even if many or all of the sessions were run in Pilot mode. It is not a matter of file size. Each session that is run is assigned the next available sequential Session Number, and there is no provision for reusing these numbers.

What you can do if you have more sessions to run using the protocol(s) in this database is to

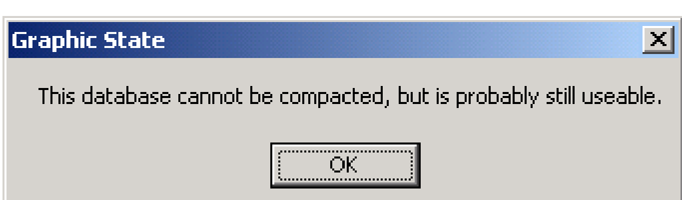
1) make a backup copy of the database (for safety and to allow future access to the data already collected)
2) use the “Clear DATA ONLY” button to delete all of the collected data while leaving the protocol definitions intact.

When you select this button, you will see the following warning:



After you verify that you have selected the correct project, Click Yes. The data will be cleared and you will see a confirmation.

After clearing the data, the program will attempt to compact the database to reclaim the unused space. It may or may not succeed. You will either see a message that the database has been compacted, or you will the following two messages.



This is only a problem if your database has reached the maximum size allowed by the operating system (approximately 1 GB). In this case, it is better to

1) export all of your protocols from the full database
2) save this database for future access
3) make a new, empty database
4) import all of the protocols into it
5) save THIS database somewhere for future use whenever you fill a database
6) restore the empty database over the full one

If you need more help

We hope this tutorial has been, and will continue to be, helpful. Call us at (610) 395-3771 or send e-mail to sales@coulbourn.com with any questions or concerns.

Sample Excel macro code

```
'
' ProcessData Macro
' Macro recorded 2/20/2002 by Martha J. Cichelli
'
' Keyboard Shortcut: Ctrl+p
'
' Summary of Protocol Logic
'  S1      UPON 61 ENTRIES, GO TO FIN
'          After variable ITI from list, GO TO either S2 or S3 from random list
'  S2      Probe trial - tone on; AFTER 60 seconds, GO TO S1
'  S3      Training cue trial - tone on; AFTER 20 seconds,GO TO S5
'  S4      Reinforcement - operate feeder, immediately GO TO S1
'  S5      Response interval - tone on; IF 1 A1-lever, GO TO S4
'                                AFTER 40 seconds, GO TO S1
'
' Desired analysis - placed in new Worksheet in raw data file
'  For each trial in each run
'    List trial number, preceding ITI, type of trial (probe or cue),
'      latency to first lever press, total length of trial,
'      number of lever presses, number of A2 responses
'

  Dim InputDataRow As Integer
  Dim OutputDataRow As Integer

  Dim LastProtocol As Integer
  Dim LastSession As Integer
  Dim LastStation As Integer
  Dim LastRun As Integer
  Dim LastSubject As String

  Dim TrialNum As Integer
  Dim StartITITime As Double
  Dim StartTrialTime As Double
  Dim InterTrialTime As Double
  Dim FirstResponseTime As Double
  Dim TimeToResponse As Double
  Dim ElapsedTrialTime As Double
  Dim TrialType As String
  Dim NumA1 As Integer
  Dim NumA2 As Integer
```

```
Dim MainSheet As String
Dim RunSheet As String

Const TimeFactor = 10 'number of units per second


Const NewITIState = 1
Const ProbeState = 2
Const TrainState = 3
Const ResponseState = 5
Const ReinfState = 4
Const FinState = -1

Const ProtocolCol = 3     'C
Const SessionCol = 4      'D
Const StationCol = 5      'E
Const RunCol = 6          'F
Const SubjectCol = 7      'G
Const TimeCol = 8         'H
Const NewStateCol = 10    'J
Const A1ResponseCol = 12  'L
Const A2ResponseCol = 13  'M

Const OutProtocolCol = 1   'A
Const OutSessionCol = 2    'B
Const OutStationCol = 3    'C
Const OutRunCol = 4        'D
Const OutSubjectCol = 5    'E
Const OutTrialCol = 6      'F
Const OutITICol = 7        'G
Const OutTypeCol = 8       'H
Const OutResponseCol = 9   'I
Const OutTrialTimeCol = 10 'J
Const OutNumA1Col = 11     'K
Const OutNumA2Col = 12     'L
```

```
Sub ProcessData()

    MainSheet = Worksheets(1).Name

    Sheets.Add      'start a new Worksheet for the output analysis
    RunSheet = Worksheets(Worksheets.Count - 1).Name
    CreateHeadings

    LastProtocol = 0
    LastSession = 0
    LastStation = 0
    LastRun = 0
    LastSubject = "      "
    TrialNum = 0

    InitializeTimers

    Worksheets(MainSheet).Activate
    InputDataRow = 2
    OutputDataRow = 2


    With ActiveSheet

      Do Until IsEmpty(Worksheets(MainSheet).Cells(InputDataRow, TimeCol).Value)

        If (Cells(InputDataRow, A1ResponseCol).Value = 1) And _
          (TrialType <> " ") Then
           NumA1 = NumA1 + 1
           If FirstResponseTime = 0 Then        'FIRST lever press recorded
              FirstResponseTime = Cells(InputDataRow, TimeCol).Value
           End If
        End If
        If (Cells(InputDataRow, A2ResponseCol).Value = 1) And _
          (TrialType <> " ") Then
           NumA2 = NumA2 + 1
        End If
        If Cells(InputDataRow, NewStateCol).Value = NewITIState Or _
          Cells(InputDataRow, NewStateCol).Value = FinState Then
           If TrialNum > 0 Then 'calculate and dump last trial
              OutputLastTrial
              InitializeTimers
           End If
        End If

        If Cells(InputDataRow, NewStateCol).Value = FinState Then
           TrialNum = 0
        End If
```

```
      If Cells(InputDataRow, NewStateCol).Value = NewITIState Then
        If NewRunStarting Then
          OutputIdInfo
          SaveIDInfo
          TrialNum = 0
        End If
        TrialNum = TrialNum + 1
        StartITITime = Cells(InputDataRow, TimeCol).Value
      ElseIf Cells(InputDataRow, NewStateCol).Value = ProbeState Then
        TrialType = "P"
        StartTrialTime = Cells(InputDataRow, TimeCol).Value
      ElseIf Cells(InputDataRow, NewStateCol).Value = TrainState Then
        TrialType = "T"
        StartTrialTime = Cells(InputDataRow, TimeCol).Value
      End If

      InputDataRow = InputDataRow + 1

     Loop

   End With

End Sub
Sub CreateHeadings()
   Worksheets(RunSheet).Activate
   With ActiveSheet
      Range("A1").Select
      ActiveCell.FormulaR1C1 = "Protocol"
      Range("B1").Select
      ActiveCell.FormulaR1C1 = "Session"
      Range("C1").Select
      ActiveCell.FormulaR1C1 = "Station"
      Range("D1").Select
      ActiveCell.FormulaR1C1 = "Run"
      Range("E1").Select
      ActiveCell.FormulaR1C1 = "Subject"
      Range("F1").Select
      ActiveCell.FormulaR1C1 = "Trial"
      Range("G1").Select
      ActiveCell.FormulaR1C1 = "ITI"
      Columns("G:G").Select
      Selection.NumberFormat = "0.0"
      Range("H1").Select
      ActiveCell.FormulaR1C1 = "Type"
      Range("I1").Select
      ActiveCell.FormulaR1C1 = "Latency"
      Columns("I:I").Select
      Selection.NumberFormat = "0.0"
      Range("J1").Select
```

```
        ActiveCell.FormulaR1C1 = "Trial duration"
        Columns("J:J").Select
        Selection.NumberFormat = "0.0"
        Range("K1").Select
        ActiveCell.FormulaR1C1 = "# A1"
        Columns("K:K").Select
        Range("L1").Select
        ActiveCell.FormulaR1C1 = "# A2"
        Columns("L:L").Select
        Selection.NumberFormat = "0"
    End With
End Sub
Sub InitializeTimers()
    StartITITime = 0
    StartTrialTime = 0
    InterTrialTime = 0
    FirstResponseTime = 0
    TimeToResponse = 0
    ElapsedTrialTime = 0
    TrialType = " "
    NumA1 = 0
    NumA2 = 0
End Sub

Sub OutputLastTrial()
    InterTrialTime = (StartTrialTime - StartITITime) / TimeFactor
    If FirstResponseTime > 0 Then
        TimeToResponse = (FirstResponseTime - StartTrialTime) / TimeFactor
    Else
        TimeToResponse = 0
    End If
    ElapsedTrialTime = (Worksheets(MainSheet).Cells(InputDataRow, TimeCol).Value _
                    - StartTrialTime) / TimeFactor
    Worksheets(RunSheet).Cells(OutputDataRow, OutTrialCol).Value = TrialNum
    Worksheets(RunSheet).Cells(OutputDataRow, OutITICol).Value = InterTrialTime
    Worksheets(RunSheet).Cells(OutputDataRow, OutTypeCol).Value = TrialType
    If TimeToResponse > 0 Then
        Worksheets(RunSheet).Cells(OutputDataRow, OutResponseCol).Value =
TimeToResponse
    Else
        Worksheets(RunSheet).Cells(OutputDataRow, OutResponseCol).Value = "*"
    End If
    Worksheets(RunSheet).Cells(OutputDataRow, OutTrialTimeCol).Value =
ElapsedTrialTime
    Worksheets(RunSheet).Cells(OutputDataRow, OutNumA1Col).Value = NumA1
    Worksheets(RunSheet).Cells(OutputDataRow, OutNumA2Col).Value = NumA2
    OutputDataRow = OutputDataRow + 1
End Sub
```

```
Function NewRunStarting()
   If Worksheets(MainSheet).Cells(InputDataRow, ProtocolCol).Value = LastProtocol
And _
      Worksheets(MainSheet).Cells(InputDataRow, SessionCol).Value = LastSession
And _
      Worksheets(MainSheet).Cells(InputDataRow, StationCol).Value = LastStation And
_
      Worksheets(MainSheet).Cells(InputDataRow, RunCol).Value = LastRun And _
      Worksheets(MainSheet).Cells(InputDataRow, SubjectCol).Value = LastSubject
Then
         NewRunStarting = False
   Else
         NewRunStarting = True
   End If
End Function

Sub OutputIdInfo()
      Worksheets(RunSheet).Cells(OutputDataRow, OutSubjectCol).Value = _
         Worksheets(MainSheet).Cells(InputDataRow, SubjectCol).Value
      Worksheets(RunSheet).Cells(OutputDataRow, OutRunCol).Value = _
         Worksheets(MainSheet).Cells(InputDataRow, RunCol).Value
      Worksheets(RunSheet).Cells(OutputDataRow, OutStationCol).Value = _
         Worksheets(MainSheet).Cells(InputDataRow, StationCol).Value
      Worksheets(RunSheet).Cells(OutputDataRow, OutSessionCol).Value = _
         Worksheets(MainSheet).Cells(InputDataRow, SessionCol).Value
      Worksheets(RunSheet).Cells(OutputDataRow, OutProtocolCol).Value = _
         Worksheets(MainSheet).Cells(InputDataRow, ProtocolCol).Value
End Sub

Sub SaveIDInfo()
   LastSubject = Worksheets(MainSheet).Cells(InputDataRow, SubjectCol).Value
   LastRun = Worksheets(MainSheet).Cells(InputDataRow, RunCol).Value
   LastStation = Worksheets(MainSheet).Cells(InputDataRow, StationCol).Value
   LastSession = Worksheets(MainSheet).Cells(InputDataRow, SessionCol).Value
   LastProtocol = Worksheets(MainSheet).Cells(InputDataRow, ProtocolCol).Value
End Sub
```