

## 1 Introduction

You will receive a lab kit (a new lunchbox kit filled with circuit boards, wires, integrated circuits, etc.) in your first lab session. Most of the labs for the remainder of the semester will require you to use the lab kit to construct simple digital circuits. The purpose of this first labkit lab is to give you some familiarity with what's provided in the kit and to give you some practice at wiring up circuits.

The material which follows is intended to introduce you to your lab kit. There are a number of precautions which you should take to avoid damaging the kit (which you will be charged for), so be sure you read this document thoroughly!

## 2 Wiring Technology

Typically, IC's are mounted on **boards** or **cards** which hold the circuits in place, and provide a means of interconnecting the chips with signal lines. In large systems, many boards may be required and a chassis is used to hold the boards in place and to provide connections between them. The board provides a rigid base on which integrated circuits and components are mounted, and it provides a means of interconnecting chips. Two common strategies for interconnecting chips on a single board are printed circuit (PC) boards and wire wrap boards. We will concentrate on wire wrap boards since this is what is provided in the lab kit.

A wire wrap board is simply a board covered with a grid of holes. Sockets are always used in wire wrap systems (see Figure 1). The pins of the chip fit into the holes on top of the socket, and make electrical contact with the square pin-like **posts**. The base of the socket sits on the board with the pins going through the holes and protruding out the other side. Thin wires wrap around the posts and provide connections between IC's. The height of the posts determine the number of wires which can be wrapped around it (typically, sockets provide enough room for two or three wires). A tool called a wire wrap tool is used to wrap the wire tightly around the post 5 or 10 times, providing up to 40 contact points to the square post.

The side of the wire wrap board with the body of the socket is called the **component side**, and the other the **wiring side**. The wire wrapped around the posts is sufficient to keep the sockets from falling out of the board so no other attachment mechanism is needed. You will use only 5 volt power supply for the your experiments this semester, so you will have to manually wire the 5 volt power pins on the component side.

The perfect wire wrap connection is one where the first wrap or first two wraps on the post (that is, the wrap closest to the board) is insulated wire, and the rest of the wrap is bare wire making contact with the post. This holds the socket on the board in a way that doesn't make electrical contact between the board and the post.

Wire wrap is often used to prototype systems before manufacturing a pc board. Wire wrap boards are not as easy to mass produce as pc boards however (this is, perhaps, a bit of an understatement...), so they are not used in high volume production. The main drawback of wire wrap boards (besides the extremely high production costs in dollars, person-hours and in time) is that they are much thicker than pc boards because of the posts protruding from the board. Thus, fewer boards can be placed into a cabinet of some given size.

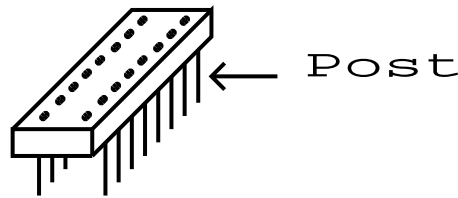


Figure 1: Wire Wrap Socket

Large systems consist of several boards, so some framework is necessary to mount them. It is important that the circuit boards are rigidly mounted into the card cage so that they cannot move around and accidentally short some part of the circuit (the box is often grounded). Some framework is also necessary to allow interboard connections. The framework for performing this function is usually referred to as the **backplane**.

A typical enclosure consists of a box, a built in power supply, perhaps some switches and lights, and a **card cage** into which the boards are mounted. A fan is also sometimes required if the circuits generate an excessive amount of heat. There should always be proper ventilation since even low power devices can build up a fair amount of heat if completely enclosed. If a fan is provided, it should circulate air over the temperature sensitive logic circuits before the less sensitive power supply. Finally, a metal shield is often placed between the power supply and the logic circuits to shield the latter from electrical noise generated by the supply.

As for connections between boards, it is **NOT** a good idea to solder or wire wrap wires from one board to another. This type of approach makes it very difficult to service the system, since boards cannot be removed or replaced without bringing out a soldering iron or a wire wrap tool. Instead, the preferred approach is to use connectors which are easily inserted or removed. The plug/wall socket on home appliances is a good example of this type of connector.

Many large computers and microcomputers are interconnected via a global bus. The computer you will construct in CS/EE3710 (if you take that course) will be constructed this way.

### 3 The CS/EE 3700 Hardware Laboratory Kit

Each of you will receive a lab kit which you will use to construct your circuits. The lab kit is a new lunchbox kit which contains:

1. A power supply,
2. An XST-2 Extender board
3. An XSA-50 Protoboard
4. A number of black wire wrap sockets for mounting chips.
5. Several packages of wire wrap wire.
6. Several jumper cables.
7. A manual wrap-unwrap tool.
8. A number of integrated circuits mounted on "bug rugs".

9. Miscellaneous items: toggle switches, push button switches, LED, and a resistor pack.

We will now discuss each of these parts in turn in order to familiarize yourself with the kit. You should have your kit in front of you while you go through this section. Identify each of the items listed above.

### 3.1 Integrated Circuits and Sockets

Lets start at the smallest components and work our way up to bigger things. First, numerous chips (the black bugs with short legs) and wire wrap sockets (the black bugs with long legs) are provided. Note that many chips have 14 pins, but no 14 pin sockets are provided. You should use 16 pin sockets for these chips.

Some precautions must be taken in handling the chips. **CMOS chips are particularly sensitive to static electricity, and can easily be destroyed if care is not taken.** When you walk across a rug, your body acts like a capacitor and collects charge. The “spark” which occurs when you touch something metal is this charge being transferred to that object. If an IC is in this current path, its circuitry will be permanently damaged. Thus, **you should never work on your kit in a room with carpets.** Further, **you should be careful in a chair with a cloth seat, as you can pick up charge as you shift positions.** Semiconductor manufacturers often have their employees wear metal rings around their wrists which are wired to ground to avoid picking up charge, however this should not be necessary here. Finally, you should do your wire wrapping *before* you insert chips into your board.

The pads or “bug rugs” on which the IC’s are mounted conduct electricity. The idea is to short the pins together so no pin is at a potential much higher than any other pin, protecting the chip from damage. To preserve this protection when you remove the chip from the pad, you should first touch a grounded object to discharge any charge you might be carrying, pick up the chip, and place it in the palm of your hand so that all of the pins contact your palm, and are (more or less) shorted together.

### 3.2 Switches

Other components are provided in your kit which are not documented in the databook and thus require further explanation. First, examine the “pushbutton switches” each of which is mounted on a 16 pin wire wrap socket. The pinout for this switch (as mounted in the 16-pin socket) is shown in Figure 2a. Note that the “top” (pins 1 and 16) corresponds to the exposed portion of the wire wrap socket. The switch is shown in its normal position. Similarly, the pinout for the toggle switch is shown in Figure 2b. The arrow on each switch indicates the direction of the switch lever.

Throughout the semester you will use mechanical switches to provide input information to your hardware. Switches either make or break an electrical connection, and can therefore be used to connect a wire to ground (logic ‘0’) or power, i.e. +5 volts (logic ‘1’). A typical circuit for using a switch to generate a ‘0’ or ‘1’ signal is shown in figure 3a. When the switch is closed, a ‘0’ is generated, and when it is open, a ‘1’ is created. Although there are other ways of achieving the same effect without using a resistor, the circuit in figure 3a works out to be a good solution.

The switch in Figure 3a may be viewed as a boolean variable which the user can set to 0 or 1 at will. It is often desirable to be able to generate such a variable *as well as its complement*. The circuit in Figure 3b achieves this effect, i.e. when one signal line is ‘0’, the other is ‘1’ and vice versa.

### 3.3 LED’s

Several LED (light emitting diode) packs are also provided in your kit. A diode is a semiconductor device which conducts current when a positive voltage  $V$  is applied biased as shown in Figure 4a. No current flows if the diode is biased in the opposite direction. It is essentially a one-way valve for electricity: current can flow in one direction but not the other. A light emitting diode is one which emits light when

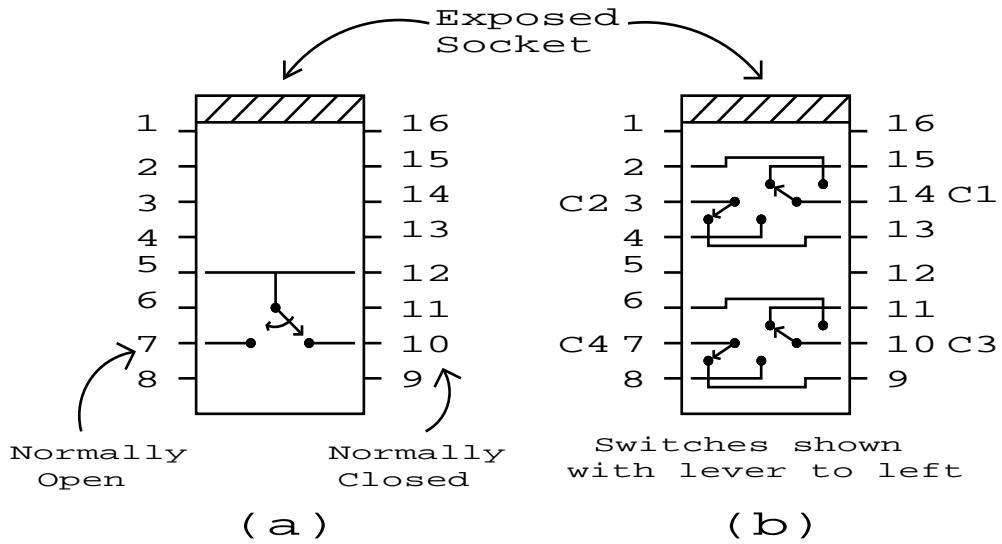


Figure 2: Switches: (a) pushbutton switch. (b) toggle switch.

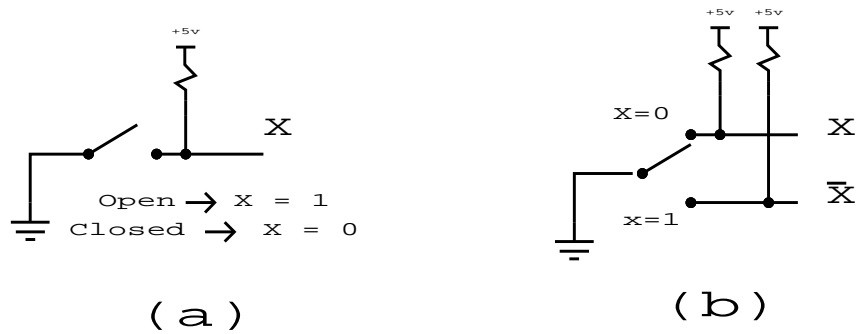


Figure 3: Circuit to generate (a) boolean variable (b) variable and its complement.

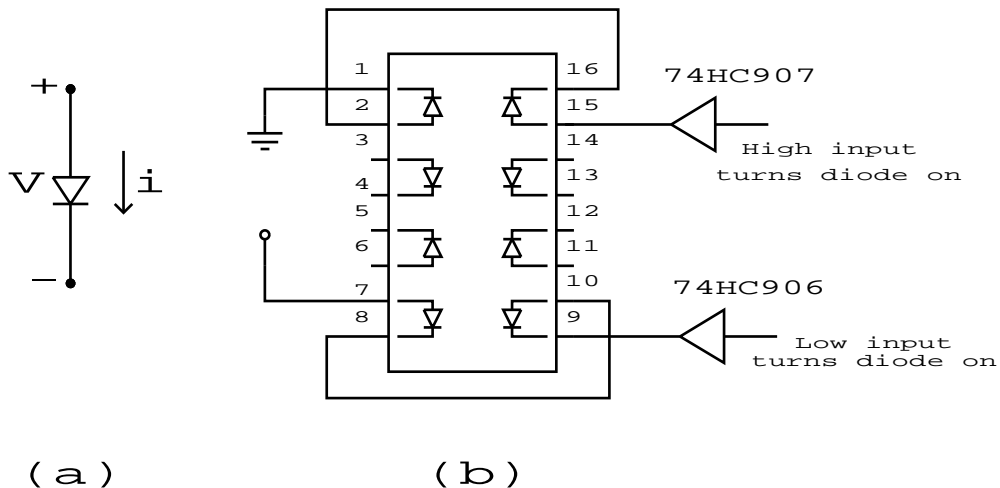


Figure 4: (a) Diode. (b) LED pack with driver circuits.

it is conducting current. Typically the more current that flows, the brighter the light (until it burns up). It turns out that the normal gates in your kit do not provide enough current to make the LEDs very bright. So, you should use 74C906 or 74C907 buffers to drive the LED's. These buffers have open-drain outputs which you'll hear more about later, but the effect is that they can drive much more current than the normal parts. In fact, they actually drive too much current. So, in order to limit the current you will use two LEDs in series (there is a voltage drop across each LED). That is, you will use the LEDs in the package in pairs. Each pair will light up one segment of LED package.

The pinout for the LED pack is shown in Figure 4b, as well as circuits to use them. The two circuits demonstrate how you can have the LED on to indicate a high or a low logic signal. Note that the LED packs are symmetrical so it is impossible to insert one upside down. Figure 4b shows how you can wire up an LED using either a 906 or a 907 driver. The rectangle that has the diode pictures inside represents the 16-pin LED package in your lab kit. You have to add the wires shown in the figure to connect the pins of the LED package in the right way to get it to light up. To be even more specific (there has been confusion in the past), to make the top LED in Figure 4b light up (this will light the entire top "bar" on the LED package) do the following:

1. Add a wire from the output of the 907 buffer to pin 15 of the LED package
2. Add a wire from pin 16 of the LED package to pin 2 of the LED package
3. Add a wire from pin 1 of the LED package to ground
4. Add a wire from somewhere else (a switch, another gate's output, etc) to the input of the 907 buffer.

Now when the signal at the input to the 907 buffer is high the LED will light up, and when that signal is low the LED will be dark.

### 3.4 Resistor Packs

A number of 16 pin resistor packs are included on the bug rug. These are meant to be used as "pull up" resistors (i.e. one end tied to 5 volts), so one terminal of all of the resistors are tied together (pin 16). The pinout is shown in Figure 5. Each resistor is 3.3K ohms. If you connect pin 16 to vdd you hen have 15 separate pull-up resistors to use in your circuit connecting to any of the other 15 pins on the package.

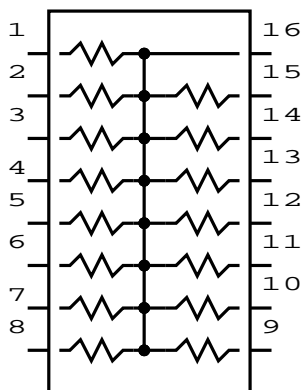


Figure 5: Resistor pack.

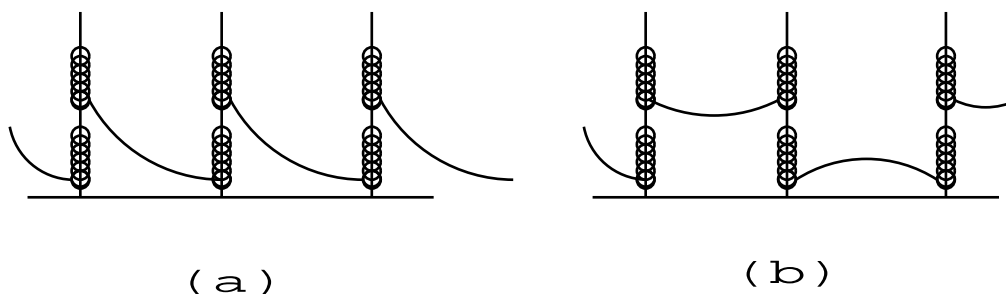


Figure 6: (a) Incorrect wiring for chain. (b) Correct wiring for a chain.

### 3.5 Wire Wrap Boards

An XST-2 Extender board is provided in your kit. For the first part of the semester you will be using the prototyping area (see Figure 8) of the XST board to wire up your circuits. If you take a close look, you will find the power (+5V) and ground (gnd) points marked (notice that there are also a 3.3V points, which you shouldn't be using for this class). If you turn the board towards the wiring side, there will be power and ground posts corresponding to the marked points. Use these posts to make your power and ground connections.

### 3.6 Wire Wrapping and Chip Extraction

A wire wrap-unwrap tool is provided in your lab kit for wiring circuits. The tool has two sides, a wire wrapping side (the longer side) and a wire unwrapping side (the shorter side). Examine the wrap tool side, which looks like a black pencil like instrument with a tip and note that there are two holes: a large one in the center (for the post), and a smaller one towards the perimeter (for the wire). Take a small piece of stripped wire (be sure to take a piece of wire wrap wire, and not one of the larger jumper cables), and place it into the *smaller* hole as far in as it will go. A small portion of the insulation should fit into the hole. Now bend the wire 90 degrees. Push the barrel of the tool into one of the soldered posts on a board (it should fit into the large hole), hold the wire in place against the board near the gun barrel, and start turning in the clockwise direction with a steady, even motion. The tool will turn, and wrap the wire around the post (see Figure 6). The portion of wire wrapped around the base of the post should have insulation on it. This helps to avoid any contact between the wire and the board.

If your wire is bunched up into a messy coil around the base of the post, then you probably held the wrap tool too tightly against the board. If there is space between the coils on the post so that you

can see the post, then you applied too little pressure. Occasionally, the wire will break, leaving part of the conductor inside the side groove of the tool. This will happen if you apply too much pressure while wrapping. The broken piece of the wire will be in the slot on the side of the tool. To remove the wire, use your fingernail or another piece of wire.

Practice wire wrapping a few wires into the board. Now take a look at the unwrap part of the tool (the shorter side), a black pencil like instrument with a tip which is somewhat similar to the wire wrap side. The tip has a single hole which fits over a post. Push the tool into a post with a wire, and turn the tool counterclockwise. The wire wrapped around the post should come undone so that you can easily remove it.

If several points must be wired together, then a “daisy chain” of connections could be used, as shown in Figure 6a. Each additional wire is added to the end of the chain. However, suppose we find that we have made a mistake, and must now remove the first (leftmost) wire. Since we must unwrap the top wire to get to the one below it, and since unwrapped wires cannot be reused reliably, the entire chain will have to be removed to remove this one wire. A better way to wire a chain of points together is shown in Figure 6b. With this approach, we need to unwrap at most three wires to remove any wire in the chain. You should always use this second method to wire several points together.

Be careful when handling your wire wrap-unwrap tool. In particular, **IT IS VERY EASY TO PERMANENTLY DAMAGE THE TOOL BY DROPPING IT. THIS WILL LEAD TO COSTLY REPAIRS WHICH YOU WILL BE EXPECTED TO PAY FOR.** Any type of slight bend in the tool will render it useless.

Finally, you should never attempt to modify or perform maintenance on the kit or any of the tools described above without first consulting with your TA. In particular, **NEVER TRY TO STRAIGHTEN THE WIRE WRAP-UNWRAP TOOL.** The tool if bent requires special care. See the DSL staff or your TA if any maintenance is required on your equipment.

## 4 Integrated Circuits

Throughout this semester you will build your hardware using integrated circuits which implement standard logic functions such as NAND gates, counters, decoders, memories, etc. It is very difficult to determine what function is performed by an integrated circuit just by looking at it (difficult, but not impossible; many manufacturers pry open their competitor’s IC’s to see how they are implementing certain functions. This process is widespread, and is referred to as **reverse engineering**). Manufacturers label IC’s with a part number to identify it, and publish descriptions of the part in **databooks**. You will be given an opportunity to read the databook description of some integrated circuits in this lab (these are available in a the handout that you should have received during your lab).

The part number usually specifies the manufacturer, the type of technology used, and the function performed. For example, in the part number MM74HC00, the MM indicates that the manufacturer is National Semiconductor, (each company has a different letter sequence, e.g. Motorola uses MC, Texas instruments uses SN, etc.). The “HC” refers to the circuit technology, in this case High-Speed CMOS. CMOS is a circuit technology traditionally known for relatively low speed but low power consumption, making it suitable for applications such as battery operated electronic calculators; recently, however, the speed of CMOS circuits has been steadily improving. Most of the chips provided in your lab kit are 74xx series CMOS chips. Finally, the 00 indicates that it is a quad 2-input NAND gate package, i.e. the chip has four circuits, each one a two input NAND gate. For this course, this is the only number of any significance since it indicates the function provided by the chip.

How do we know that 74HC00 refers to a NAND gate? The databook provides this information. Each manufacturer publishes a **databook** which is a catalog indicating the logic function performed by each part the company manufactures. The databook also includes the **pinouts** for the part, i.e. which pins are

the inputs to the first NAND gate, which is the output, etc., as well as timing and electrical specifications for the chip (how fast it is, what are the input/output voltage and current levels, etc.) For this course, you can treat logic gates as black boxes which perform standard logic functions.

A small notch is usually placed at the top of each integrated circuit to denote the top. Alternatively, a small mark may be placed in the upper left hand corner of the chip. When the chip is oriented this way, pin 1 refers to the pin in the upper left hand corner, pin 2 to the one below it, and so on going down the left edge of the chip, and then back *up* the right edge. A 14-pin chip has 7 pins on each edge. By convention, the pin on the lower left hand corner of the chip (pin 7 for a 14 pin chip) is usually connected to ground, and the pin in the upper right hand corner (pin 14 on 14-pin chip) is used for power. This is not always the case however, so you should always check the datasheets to be sure.

## 5 Viewdraw Component Libraries

There are two main libraries from which you should select components to draw your schematics in Viewdraw. The **devices** library that contains the 74xx series gates, and the **labkit** library that contains some special components that correspond to the switches and LEDs in your lab kits.

The gates in the **devices** library are numbered according to standard 74xx series practice. Note that there are two versions of many gates. For example, a 74hc00 NAND gate can be referred to as 74hc00.1 or as 74hc00.2 These gates are all the same two-input gate, but have different symbols. The .1 form is the standard distinctively shaped version of a NAND gate, and the .2 version will be the dual form drawn as an OR gate with inversion on the inputs. **Always use the gate whose shape corresponds to your intended function.** If you are ANDing two signals together, use an AND-form gate. If you are ORing, use an OR-form.

There are also some circuits in your labs that do not correspond to components in the devices library. These components are found in the **labkit library**. In particular, the toggle switch, push-button switch, 74c906 and 74c907 LED drivers, the LEDs, and the pull-up resistor components are available in the **labkit library**.

The 74c906 and 74c907 chips in the (labkit) library are “open-drain” buffers. That means that they are constructed such that the output transistor of the device is unconnected (“open”) on one of its connections (the “drain” connection). You can see this on the data sheet. The 74c906 has an open-drain N-type transistor on the output so it can pull its output low through that transistor, but has no means for driving the output high. In order to use this gate as a buffer, you must connect an external pull-up resistor to the output. The 74c907 is the opposite. Because its open-drain output transistor is a p-type device, it can pull its output high, but must have an external pull-down resistor in order to allow the output to be low. These can be used to build open-drain gates where many gates are connected to a common pull-up (in the 906 case) device. In this case any of the 906’s could pull the common wire low, and only if none of them are pulling low will the external pull-up raise the common wire high. This is called an open-drain AND connection.

These 906 and 907 devices are also used as LED drivers because they supply more current than the other gates in your kit. The LEDs require that current to light up. The LEDs are shown as symbols that look like the LEDs in Figure 4. There are two symbols, one connected as in the top of Figure 4 and meant to be connected to a 74hc907 in your schematic, and the other connected as in the bottom of the figure and meant to be connected to a 74hc906. Note that although they are drawn as two different devices, they are actually the same LEDs, just connected differently. In your simulation, however, you should make sure that you use the correct LED to match with your selected LED driver in your Viewdraw schematic. The output from the LED symbols will be 1 when the LED is lit, and 0 when the LED is off.

In the labkit library you will also find toggle-switch and pb-switch (push-button switch) components. These switches behave in the simulator in the same way as the real switches behave in your circuits.

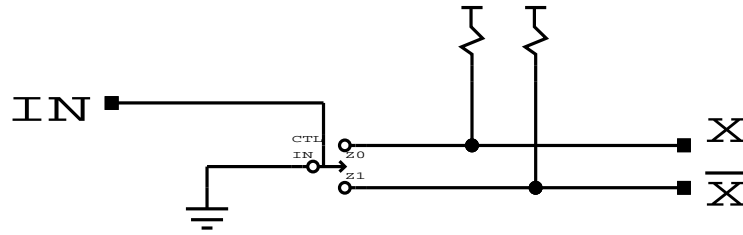


Figure 7: Toggle Switch Example from Viewdraw.

The difference is that in order to simulate them, you need some way to indicate the physical switching of the switch. In the case of the switches in the (labkit) library, there are three connections that are the connections to the switches, and a fourth control input that you can use in the simulator to control the switches. In the toggle switch, for example, a 0 applied to the control input means that the common input to the switch is connected to the Z0 output. A 1 on the control input means that the common pin is connected to the Z1 output. In the pushbutton case, a control input of 0 means that the button is not pressed and the connection is made as shown in the symbol. If the control input is 1, then the button is pressed and the common connection of the switch is connected to the bottom (Z1) output of the pushbutton.

To use the switches to generate high and low inputs in your circuit, you need to connect pull-up resistors, and a ground connection, as shown in Figure 3. To include these pull-up resistors and ground connections in your schematic, there are components in the labkit library called “pullup” and “gnd.”. The pullup cell is a resistor connected to power, and the gnd cell is a symbol that connects a net to ground. As you might guess, there are also resistors to ground (which you probably won’t need this semester) in “pulldown” and a power connection called “pwr.”

The circuit in Figure 2 as it would be drawn in Viewdraw is shown in Figure 7. Note that if you do not include the pullup and ground connections as shown, your switch will not simulate correctly!

## 6 A Design Methodology

A haphazard design strategy usually leads to hardware which is difficult to modify and/or debug. Thus, we will now discuss a methodology (i.e. a procedure) which you should follow in constructing your circuits.

You should proceed through the following steps in designing and constructing your hardware:

1. Prepare a high level description of the major components of your design. Components at this level might include a finite state machine, a circuit for performing serial addition, etc. Get a clear picture in your mind of exactly what function will be performed by each module, how they are interconnected, and functionally, what control and data signals must flow between them. In a finite state machine for example, the description would include what operations the machine performs, what inputs are required (e.g. to start the machine through a sequence of steps), and what outputs are generated. If the circuits are sufficiently simple (as in this week’s lab), then this step may be omitted.
2. Prepare a detailed logic design of your circuits, using the databook to identify precisely what functions are performed by each chip. Use Viewdraw or Schematic Editor to draw the schematics for your design. Each block in the design should correspond to a component available in your lab kit. Make sure you have all of the components you need in your design. In the earlier labs, use components with just enough functionality to perform the task at hand (e.g. don’t use an 8 bit register if you only need 2 bits). You should save components with additional functionality for later labs.

Once you have the detailed logic design entered into the schematic, use Viewsim or Logic Simulator to simulate the circuit. Simulating before constructing the circuit can track down many errors before it becomes necessary to re-wire circuits on the wire wrap board! **Once you have completed this much of the procedure, you have completed the prelab and are ready to have your design checked by a TA.** Make sure your documentation at this point is neat and easy to understand.

3. Figure out a placement of your chips on the board. You can do this easily by using an on-line version of the board diagram for this purpose, but only in Viewdraw. There are no Foundation symbols for this yet... If you're using Powerview, there is a symbol called `ww-board` in the labkit library that is a representation of the wire wrap board. There are also symbols that represent each of the chip types that you will use in your labs. Each of these symbols is named something starting with the word "chip." For example, `chip16` is a 16-pin chip socket, and `chiptoggle` is a representation of a toggle-switch package. You can use these to indicate where on the wire wrap board you are planning on putting each of the chips in your implementation. You can make a schematic called "board-layout" and use the `ww-board` and `chip` symbols to show where you are planning on putting the parts in your implementation. Note that this looks a little ugly in the printed version. The on-line version uses different colors to make things look better.

Sockets are oriented 90 degrees from the wire wrap sockets used for making contact with the edge connector. You should be able to fit 10 columns and 4 or 5 rows of 16 pin sockets on each board. When looking at the component side of the board with the edge connector on the bottom edge, columns of chips are labeled A, B, C, ... from left to right, and rows 1, 2, 3, ... from *bottom to top*. Thus, chip 1A is in the lower left hand corner of the board. Each chip should be oriented so pin 1 is in the upper left hand corner. Note that chip 1A corresponds to the lower *right* hand position of the board when looking on the wiring side. The pinout of each socket will similarly be affected. You may find it helpful to make a list of all of the components you need (don't forget switches, LED's, resistors, etc.), find an appropriate socket with enough pins for each part, and then Circuit demo due during your lab during the week of February 25th

(labs start Wednesday February 27th, first day back from Olympic break! )

physically place these sockets into a board. This will avoid such mistakes as putting too many chips in a column or not accounting for 24-pin chips which require a wide socket (and use two columns).

4. Indicate on your logic diagram the chip position of each gate and the pin number corresponding to each input/output (Pin numbers can be found on the data sheet for the part). Note that if you're using Powerview, some of the parts from the (labkit) and (devices) libraries have pin numbers placed automatically in the schematic when you place components in ViewDraw. (This doesn't work in Foundation) You can change these pins by changing the "slot" of the part. For example, to change which of the four possible sets of pin numbers are shown for a NAND gate, use the `Change→Slot` command in ViewDraw to change whether the NAND gate in the 1, 2, 3, or 4 slot is being used. The end result is that just by looking at the logic diagram, you should be able to locate the location of any signal on your board.
5. Indicate in parenthesis on your logic diagrams the actual position of the pin on the wire wrap board. Note that columns of posts on each board are labeled by letters, and rows by numbers. Be careful with the 14 pin packages because they will be placed in 16 pin sockets, meaning 2 posts are not connected to anything. As a convention, always leave pins 1 and 16 of the socket unconnected, implying that pin  $i$  on the chip corresponds to pin  $i+1$  of the socket.
6. Make photocopies of your logic diagrams.
7. Prepare a wire list for your circuit, which you will use to wire wrap your board. Be sure to include

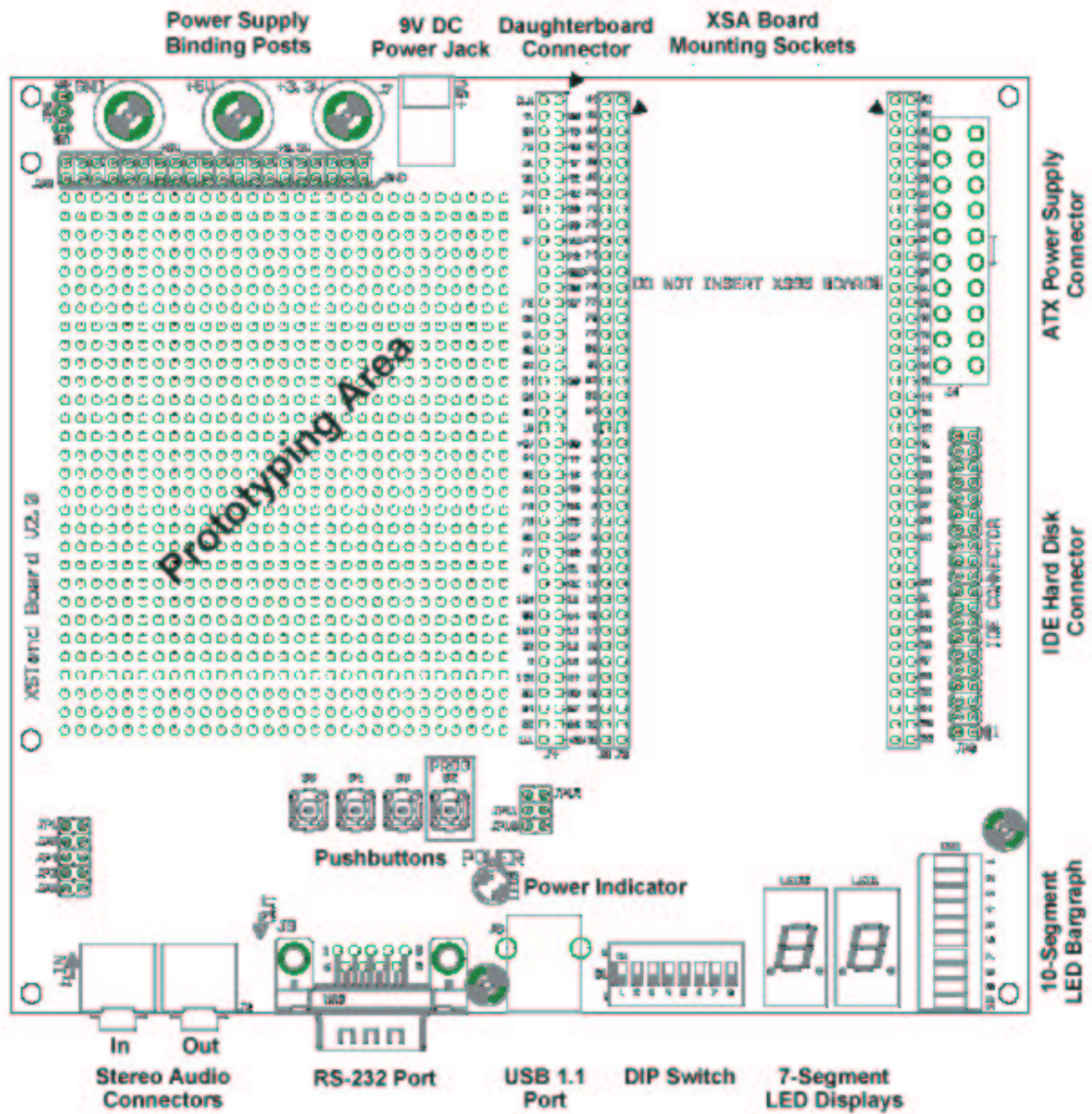


Figure 8: A Picture of the XST-2 Extender Board

power and ground connections for each chip. Each entry of the list should include the coordinates of two board positions (in parenthesis on the diagram). You may find it useful to take a copy of your logic diagram, and cross off pin numbers as you enter them in your wire wrap list. This way, you will be able to easily determine which pins have not yet been entered on the list.

8. Insert the sockets into the board and wire up power and ground connections. Be careful if some 16 pin sockets contain 14 pin chips! You might want to wire some more dummy wires to these unused posts as a reminder that they are not to be used.
9. Check your power and ground wiring. Make sure that the power and ground pins of the socket connect to the appropriate points on the edge connector. The easiest way to do this test is to plug the board (*without chips!*) into the chassis, turn on power, and check the appropriate points with a logic probe, volt meter, or oscilloscope *from the component side of the board*. **Note: If the light does not go on when you turn on power, turn power off immediately as this could indicate a short between power and ground (see below).** Taking a few minutes to check that each chip gets power and ground will save you countless hours of headaches later on, as these sorts of mistakes are not easy to track down. **Further, connecting the ground pin of the chip to +5 volts and the power pin to ground will probably destroy the chip**, so you'll want to make sure you get it right.
10. Use your wire list to wire wrap the rest of the board.
11. Before the chips have been inserted into the board, check your wiring from the *component* side of the board. Use a copy of your logic diagram and a highlight pen (rather than the wiring list) to do this check, marking off lines as they are checked. This will detect errors made in generating the wiring list as well as wiring errors. Perform this test with an ohmmeter, or if you don't have one, use your logic probe and a long piece of wire with one end wrapped to ground. In the latter case, insert the board into the edge connector and turn power on. To test for a connection from point A to point B, first connect the probe to point A. It should not produce any signal (if it does, you have erroneously connected the line to power or ground). Now attach the grounded wire to point B. The logic probe should register a logic zero. If you're using an ohm meter, note that it will **not** give useful readings after the chips have been inserted. **Wiring errors are among the most frustrating, so you should always check your wiring before inserting any chips into the board.** A few minutes now will often save you hours of headaches and frustrations later on.
12. Insert the chips into the socket, and begin debugging the circuit.
13. If you didn't check your wiring and your circuit doesn't work because of a wiring error, don't say we didn't tell you so!
14. Make all changes on a copy of your logic diagram, and when the circuit works, update the originals and the wire list.

## 7 Some Final Notes

There are some miscellaneous facts you should be aware of about the lab kit. First, if you plug the XST in and the light on the power switch does **NOT** come on, then this probably means you have a short circuit from power to ground. Turn the power off immediately, and test the board to locate the short.

If you turn power off and the power light remains on for a few seconds don't be alarmed. The circuitry on the boards stores up charge which keeps the light on until the charge is dissipated.

When you wire your board, note that the orientation of the chips is reversed from when you look at them from the component side. From the wiring side, pin number 1 is in the upper right hand corner. Also

be sure you insert the chip with the notch at the top, or the dot over pin 1 on the upper left hand corner. **Chips inserted upside down will usually be permanently damaged.**

In all labs you should pack your circuits onto the wire wrap board as tightly as is reasonable so that you don't waste space between chips. Don't spread your chips all over the board.

## **7.1 Rules on Parts and Wire**

Throughout this semester, you will **NOT** be allowed to use any parts beyond what is provided in the lab kit. You are probably doing something wrong if you run out of parts. You may not use parts from your partner's or anyone else's lab kit. You will be allowed to replace damaged parts however. You must not swap wire wrap boards with anyone else. **You will not receive any credit for work done on a wire wrap board which was not checked out to you.**

You may purchase new wire by buying a card from the bookstore, and presenting it to a DSL staff member who will punch out holes in it in exchange for new wire. Alternatively, you may exchange unopened packages of wire of one length for packets of a different length. You will probably find you will need much more wire of the shorter lengths than the longer lengths. You are also free to exchange wire with other students to obtain lengths you need. There is also a box of free wire in the DSL that you can use at your own risk. There is a small chance that the bare wire in the open packets will corrode and therefore not make good electrical contact. Most of this free wire will be fine though.