XESS Corporation

2608 Sweetgum Drive
Apex NC 27502
Toll-free: 800-549-9377
International: 919-387-0076
FAX: 919-387-1302

XS95 Board V1.3 User Manual

How to install, test, and use your new XS95 Board

RELEASE DATE: 9/21/2001

1 Preliminaries

Getting Help!

Here are some places to get help if you encounter problems:

- If you can't get the XS95 Board hardware to work, send an e-mail message describing your problem to help@xess.com or submit a problem report at http://www.xess.com/reqhelp.html. Our web site also has
    - answers to frequently-asked-questions,
    - example designs for the XS Boards,
    - application notes,
    - a place to sign-up for our email forum where you can post questions to other XS Board users.
- If you can't get your XILINX Foundation software tools installed properly, send an e-mail message describing your problem to hotline@xilinx.com or check their web site at http://support.xilinx.com.

Take notice!!

- The XS95 Board requires an external power supply to operate! It does not draw power through the downloading cable from the PC parallel port.
- If you are connecting a 9VDC power supply to your XS95 Board, please make sure the center terminal of the plug is positive and the outer sleeve is negative.
- The V1.3 version of the XS95 Board now uses a programmable oscillator with a default frequency of 50 MHz. You must reprogram the oscillator if you want to use another frequency. The procedure for doing this is described on page 7.

Packing List

Here is what you should have received in your package:

- an XS95 Board;
- a 6' cable with a 25-pin male connector on each end;
- an XSTOOLs CDROM with software utilities and documentation for using the XS95 Board.

2 Installation

Installing the XSTOOLs Utilities and Documentation

XILINX currently provides the Foundation tools for programming their FPGAs and CPLDs. Any recent version of XILINX software should generate bitstream configuration files that are compatible with your XS95 Board. Follow the directions XILINX provides for installing their software. You can get additional help at http://xup.msu.edu/license/index.htm.

XESS Corp. provides the additional XSTOOLs utilities for interfacing a PC to your XS95 Board. Run the SETUP.EXE program on the XSTOOLs CDROM to install these utilities.

Applying Power to Your XS95 Board

You can use your XS95 Board in two ways, distinguished by the method you use to apply power to the board.

Using a 9VDC wall-mount

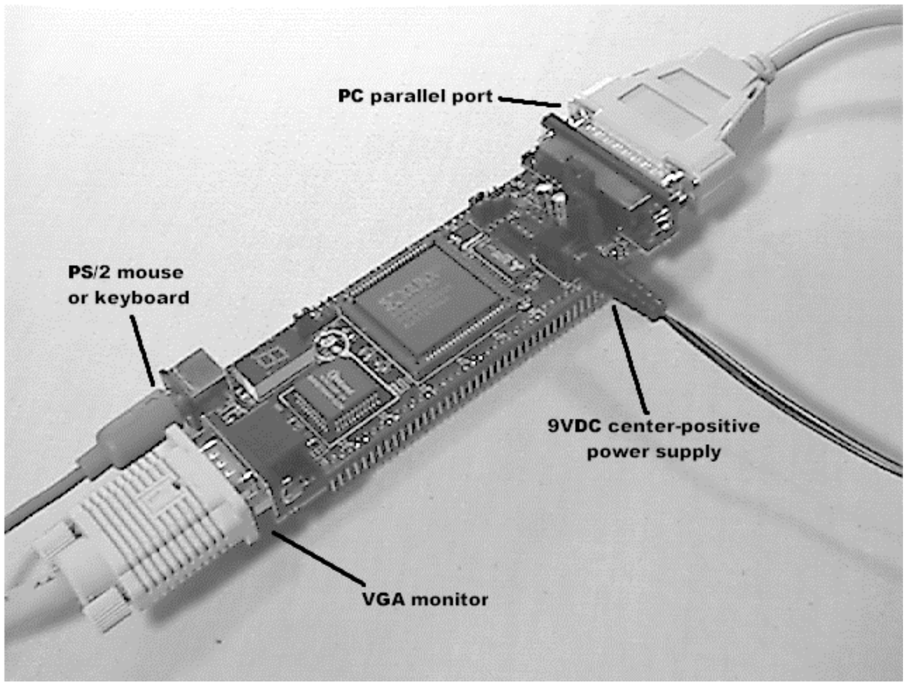
You can use your XS95 Board all by itself to experiment with logic and microcontroller designs. Just place the XS95 Board on a non-conducting surface as shown in Figure 1. Then apply power to jack J9 of the XS95 Board from a 9V DC wall transformer with a 2.1 mm female, center-positive plug. (See Figure 2 for the location of jack J9 on your XS95 Board.) The on-board voltage regulation circuitry will create the voltages required by the rest of the XS95 Board circuitry.

Solderless Breadboard Installation

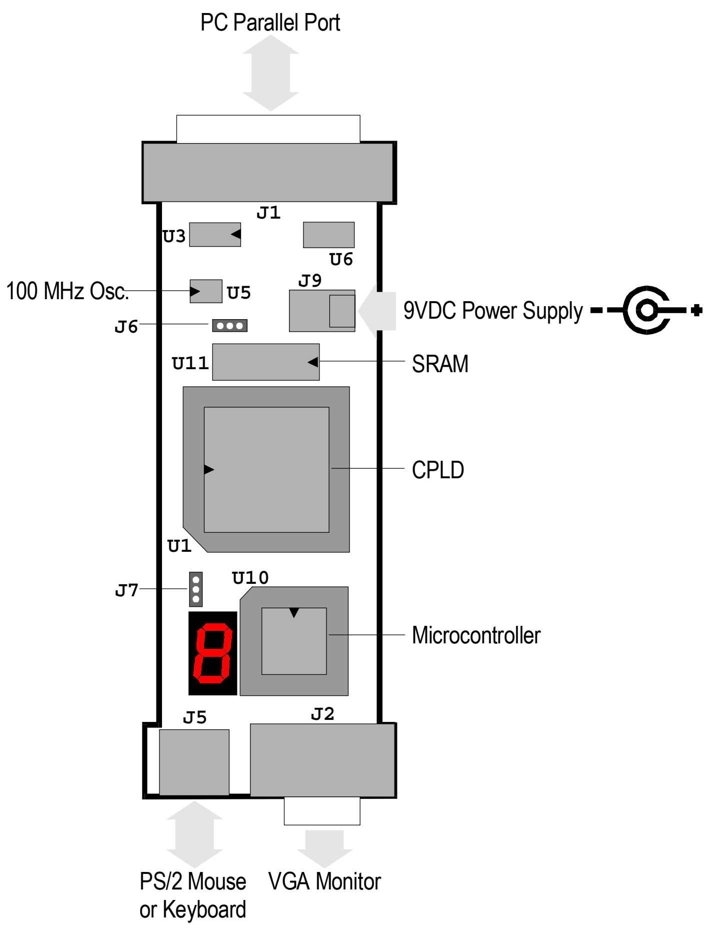
The two rows of pins from your XS95 Board can be plugged into a solderless breadboard with holes spaced at 0.1" intervals. (One of the A.C.E. breadboards from 3M is a good choice.) Once plugged in, all the pins of the CPLD, microcontroller, and SRAM are accessible to other circuits on the breadboard. (The numbers printed next to the rows of pins on your XS95 Board correspond to the pin numbers of the CPLD.) Power can still be supplied to your XS95 Board though jack J9, or power can be applied directly through several pins on the underside of the board. Just connect +5V and ground to the following pins for your XS95 Board.

• Table 1: Power supply pins for the XS95 Board.

XS Board Type	GND Pin	+5V Pin
XS95-108 V1.3	49	78
XS95-108+ V1.3	49	78



• Figure 1: External connections to the XS95 Board.



• Figure 2: Arrangement of components on the XS95 Board.

Connecting a PC to Your XS95 Board

The 6' cable included with your XS95 Board connects it to a PC. One end of the cable attaches to the parallel port on the PC and the other connects to the female DB-25 connector (J1) at the top of the XS95 Board as shown in Figure 1.

Connecting a VGA Monitor to Your XS95 Board

You can display images on a VGA monitor by connecting it to the 15-pin J2 connector at the bottom of your XS95 Board (see Figure 1). You will have to download a VGA driver circuit to your XS95 Board to actually display an image. You can find an example VGA driver at http://www.xess.com/ho03000.html.

Connecting a Mouse or Keyboard to Your XS95 Board

You can accept inputs from a keyboard or mouse by connecting it to the J5 PS/2 connector at the bottom of your XS95 Board (see Figure 1). You can find an example keyboard driver at http://www.xess.com/ho03000.html.

Setting the Jumpers on Your XS95 Board

The default jumper settings shown in Table 2 configure your XS95 Board for use in a logic design environment. You will need to change the jumper settings only if you are:

- reprogramming the clock frequency on your XS95 Board (see page 7),
- executing microcontroller code from internal ROM instead of the external SRAM on the XS95 Board. (You will have to replace the ROMless microcontroller on the XS95 Board with a ROM version to use this feature.)

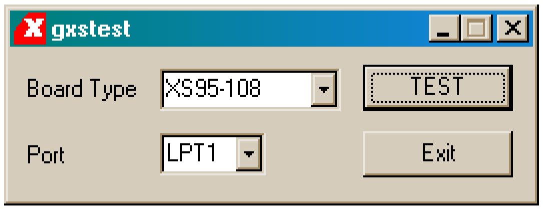
• Table 2: Jumper settings for XS95 Board.

Jumper	**Setting**	**Purpose**
J6	2-3 (osc) (default)	The shunt should be installed on pins 2 and 3 (osc) during normal operations when the programmable oscillator is generating a clock signal.
	1-2 (set)	The shunt should be installed on pins 1 and 2 (set) when the programmable oscillator frequency is being set.
J7	1-2 (ext) (default)	The shunt should be installed on pins 1 and 2 (ext) if the microcontroller program is stored in the external SRAM (U11) of the XS95 Board.
	2-3 (int)	The shunt should be installed on pins 2 and 3 (int) if the program is stored internally in the ROM of the microcontroller.

Testing Your XS95 Board

Once your XS95 Board is installed and the jumpers are in their default configuration, you can test the board using the GUI-based GXSTEST utility as follows.

You start GXSTEST by clicking on the GXSTEST icon placed on the desktop during the XSTOOLS installation. This brings up the window shown below.



Board Type XS95-108
Port LPT1

Next you select the parallel port that your XS95 Board is connected to from the Port pulldown list. GXSTEST starts with parallel port LPT1 as the default, but you can also select LPT2 or LPT3 depending upon the configuration of your PC.

After selecting the parallel port, you select the type of XS95 Board you are testing from the Board Type pulldown list. Then click on the TEST button to start the testing procedure. GXSTEST will configure the CPLD to perform a test procedure on your XS95 Board. After several seconds you will see a **O** displayed on the LED digit if the test completes successfully. Otherwise an **E** will be displayed if the test fails. A status window will also appear on your PC screen informing you of the success or failure of the test.

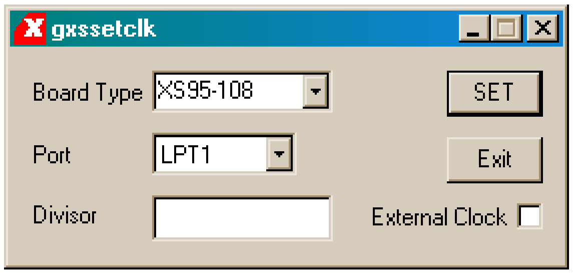
If your XS95 Board fails the test, you will be shown a checklist of common causes for failure. If none of these causes applies to your situation, then test the XS95 Board using another PC. In our experience, 99.9% of all problems are due to the parallel port. If you cannot get your board to pass the test even after taking these steps, then contact XESS Corp for further assistance.

Programming Your XS95 Board Clock Oscillator

The XS95 Board has a 100 MHz programmable oscillator (a Dallas Semiconductor DS1075Z-100). The 100 MHz master frequency can be divided by factors of 1, 2, ... up to 2052 to get clock frequencies of 100 MHz, 50 MHz, ... down to 48.7 KHz, respectively. The divided frequency is sent to the rest of the XS95 Board circuitry as a clock signal.

The divisor is stored in non-volatile memory in the oscillator chip so it will resume operation at its programmed frequency whenever power is applied to the XS95 Board. You can store a particular divisor into the oscillator chip by using the GUI-based GXSSETCLK utility as follows.

You start GXSSETCLK by clicking on the GXSSETCLK icon placed on the desktop during the XSTOOLS installation. This brings up the window shown below.



gxssetclk

Board Type: XS95-108

Port: LPT1

Divisor:

External Clock: ☐

Your next step is to select the parallel port that your XS95 Board is connected to from the Port pulldown list. GXSSETCLK starts with parallel port LPT1 as the default, but you can also select LPT2 or LPT3 depending upon the configuration of your PC. Then select the type of XS95 Board from the Board Type pulldown list.

Next you enter a divisor between 1 and 2052 into the Divisor text box and then click on the SET button. Then follow the sequence of instructions given by GXSSETCLK for moving shunts and removing and restoring power during the oscillator programming process. At the completion of the process, the new frequency will be programmed into the DS1075.

An external clock signal can be substituted for the internal 100 MHz oscillator of the DS1075. Checking the External Clock checkbox will enable this feature in the programmable oscillator chip. If this option is selected, you are then responsible for providing the external clock to the XS95 Board through pin 64.

3 Programming

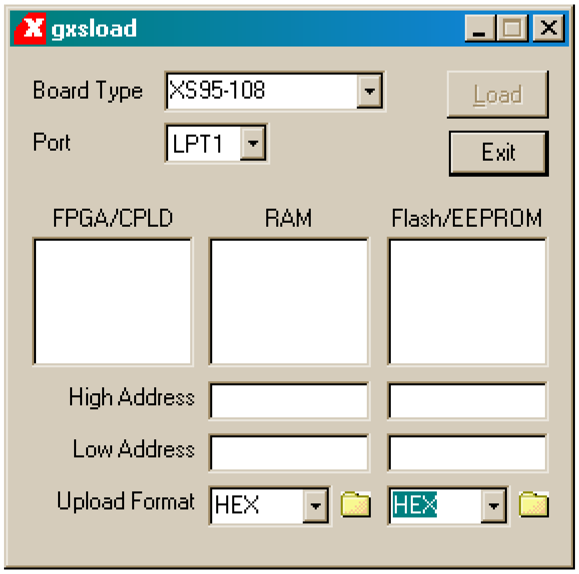
This section will show you how to download a logic design from a PC into your XS95 Board.

Downloading Designs into Your XS95 Board

During the development and testing phases, you will usually connect your XS95 Board to the parallel port of a PC and download your circuit each time you make changes to it. You can download a CPLD design into your XS95 Board using the GXSLOAD utility as follows.

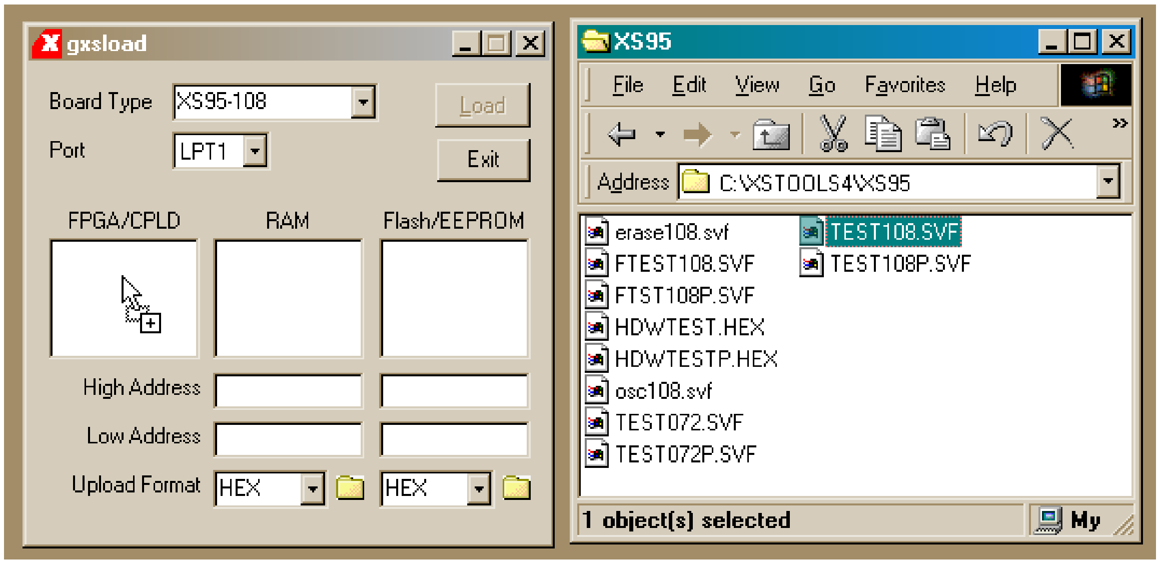
You start GXSLOAD by clicking on the GXSLOAD icon placed on the desktop during the XSTOOLS installation. This brings up the window shown below.

Next you select the parallel port that your XS95 Board is connected to from the Port pulldown list. GXSTEST starts with parallel port LPT1 as the default, but you can also select LPT2 or LPT3 depending upon the configuration of your PC. Then select the type of XS95 Board you are using from the Board Type pulldown list.

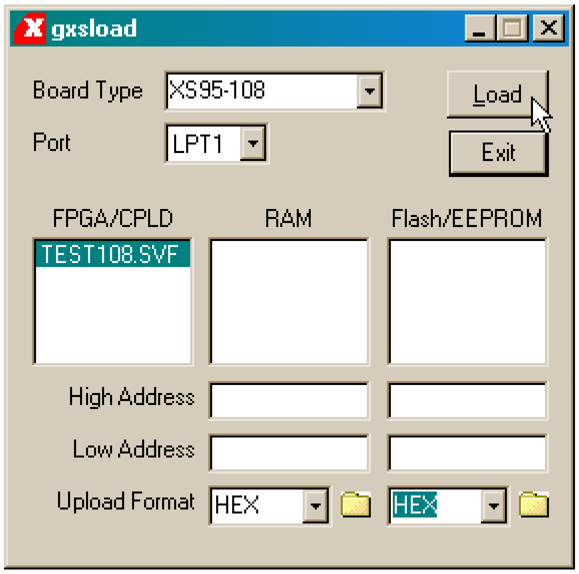


Board Type: XS95-108
Port: LPT1
Upload Format: HEX
Upload Format: HEX

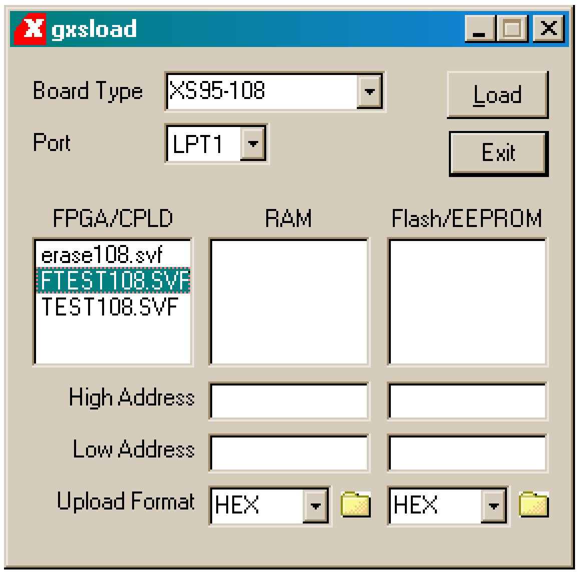
After setting the board type and parallel port, you can download .SVF files to the CPLD on your XS95 Board simply by dragging them to the FPGA/CPLD area of the GXSLOAD window as shown below.



Once you release the left mouse button and drop the file, the highlighted file name appears in the FPGA/CPLD area and the Load button in the GXSLOAD window is enabled. Clicking on the Load button will begin sending the highlighted file to the XS95 Board through the parallel port connection. .SVF files contain configuration bitstreams that are loaded into the CPLD. GXSLOAD will reject any non-downloadable files (ones with a suffix other than .BIT or .SVF). During the downloading process, GXSLOAD will display the name of the file and the progress of the current download.



You can drag & drop multiple files into the FPGA/CPLD area. Clicking your mouse on a filename will highlight the name and select it for downloading. Only one file at a time can be selected for downloading.


gxsload

Board Type: XS95-108

Port: LPT1

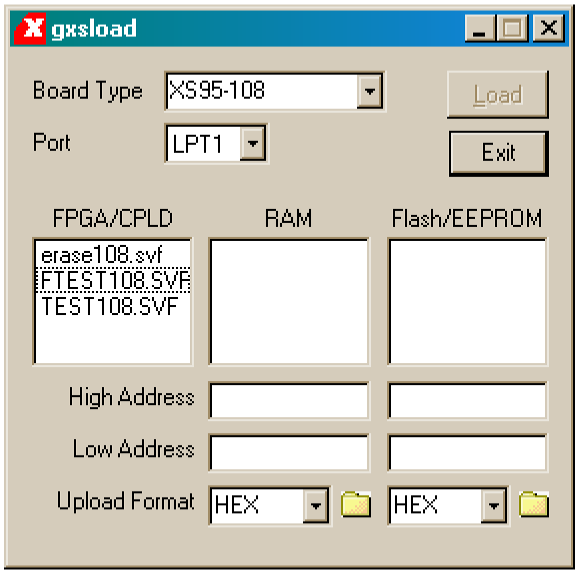
FPGA/CPLD: erase108.svf, FTEST108.SVF, TEST108.SVF

High Address

Low Address

Upload Format: HEX, HEX

Double-clicking the highlighted file will deselect it so no file will be downloaded Doing this disables the Load button.


gxsload

Board Type: XS95-108

Port: LPT1

FPGA/CPLD: erase108.svf, FTEST108.SVF, TEST108.SVF

High Address

Low Address

Upload Format: HEX, HEX

Downloading and Uploading Data to/from the RAM in Your XS95 Board

The XS95 Board contains 32 or 128 KBytes of RAM whose contents can be downloaded and uploaded by GXSLOAD. This is useful for initializing the RAM with data for use by the CPLD and then reading the RAM contents after the CPLD has operated upon it. The RAM is loaded with data by dragging & dropping one or more .EXO, .MCS, .HEX, and/or .XES files into the RAM area of the GXSLOAD window and then clicking on the Load button. This activates the following sequence of steps:

1. The CPLD on the XS95 Board is reprogrammed to create an interface between the RAM device and the PC parallel port.

2. The contents of the .EXO, .MCS, .HEX or .XES files are downloaded into the RAM through the parallel port. **The data in the files will overwrite each other if their address ranges overlap.**

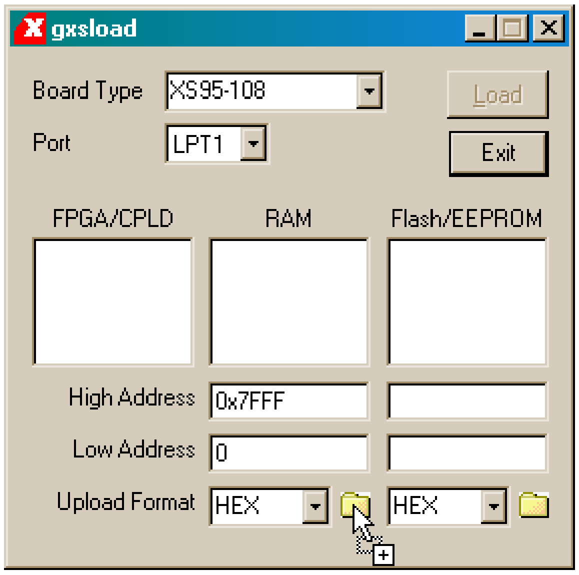
3. After the data is downloaded to the RAM, any highlighted bitstream file in the FPGA/CPLD area is downloaded into the CPLD on the XS95 Board. Otherwise the CPLD remains configured as an interface to the RAM.

You can also examine the contents of the RAM device by uploading it to the PC. To upload data from an address range in the RAM, type the upper and lower bounds of the range into the High Address and Low Address fields below the RAM area, and select the format in which you would like to store the data using the Upload Format pulldown list. Then click on the file icon and drag & drop it into any folder. This activates the following sequence of steps:

1. The CPLD on the XS95 Board is reprogrammed to create an interface between the RAM device and the PC parallel port.

2. The RAM data between the high and low addresses (inclusive) is uploaded through the parallel port.

3. The uploaded data is stored in a file named RAMUPLD with an extension that reflects the file format.



Board Type XS95-108
Port LPT1
High Address 0x7FFF
Low Address 0
Upload Format HEX HEX

4 Programmer's Models

This section discusses the organization of components on the XS95 Board and introduces the concepts required to create applications that use both the microcontroller and the CPLD. Building CPLD-based designs is covered in detail in the *Pragmatic Logic Design* online text found at http://www.xess.com/pragmatic-2_1.html. Designs that couple the operations of the CPLD with the microcontroller are discussed in the online document http://www.xess.com/appnotes/an-103100-ucfpga.pdf.

Microcontroller + CPLD Design Flow

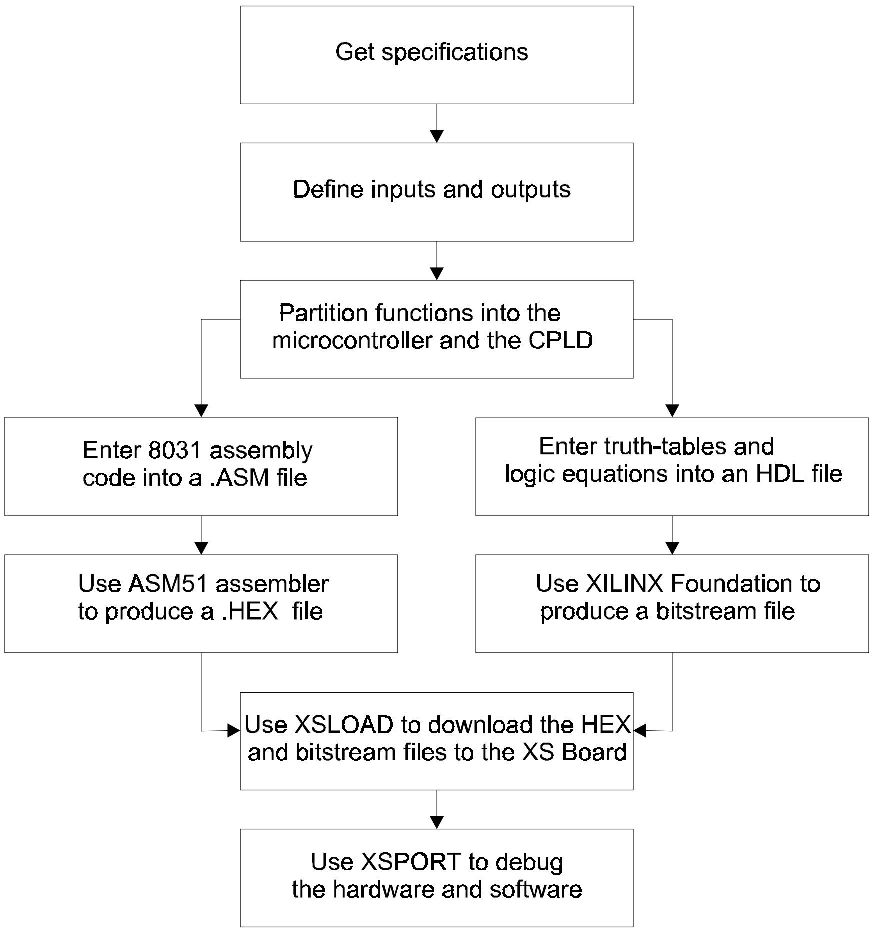
The basic design flow for building microcontroller+CPLD applications is shown in Figure 3. Initially you have to get the specifications for the system you are trying to design. Then you have to determine what inputs are available to your system and what outputs it will generate.

At this point, you have to partition the functions of your system between the microcontroller and the CPLD. Some of the input signals will go to the microcontroller, some will go to the CPLD, and some will go to both. Likewise, some of the outputs will be computed by the microcontroller and some by the CPLD. There will also be some new intra-system inputs and outputs created by the need for the microcontroller and the CPLD to cooperate.

In general, the CPLD will be used mainly for low-level functions where signal transitions occur more frequently and the control logic is simpler. A specialized serial transmitter/receiver would be a good example. Conversely, the microcontroller will be used for higher-level functions where the responses occur less quickly and the control logic is more complex. Reacting to commands passed in by the receiver is a good example. Once the design has been partitioned and you have assigned the various inputs, outputs, and functions to the microcontroller and the CPLD, then you can begin doing detailed design of the software and hardware. For the software, you can use your favorite editor to create a .ASM assembly-language file and assemble it with ASM51 to create a .HEX file for the microcontroller on the XS95 Board. For the CPLD hardware portion, you will enter truth-tables and logic equations into a .ABL or .VHDL file and compile it into an .SVF bitstream file using the XILINX Foundation software.

You can download the .HEX program file and the .SVF bitstream file to the XS95 Board using the XSLOAD program. XSLOAD stores the contents of the .HEX file into the SRAM on the XS95 Board and then it reconfigures the CPLD by loading it with the bitstream file.

When the XS95 Board is loaded with the hardware and software, you need to test it to see if it really works. The answer usually starts as "No" so you need a method of injecting test signals and observing the results. XSPORT is a simple program that lets you send test signals to the XS95 Board through the PC parallel port. You can trace the reaction of your system to signals from the parallel port by programming the microcontroller and the CPLD to output status information on the LED digit (much like placing "printf" statements in your C language programs). This is admittedly crude but will serve if you don't have access to a programmable stimulus generator or logic analyzer.


Get specifications
Define inputs and outputs
Partition functions into the microcontroller and the CPLD
Enter 8031 assembly code into a .ASM file
Enter truth-tables and logic equations into an HDL file
Use ASM51 assembler to produce a .HEX file
Use XILINX Foundation to produce a bitstream file
Use XSLOAD to download the HEX and bitstream files to the XS Board
Use XSPORT to debug the hardware and software

• Figure 3: CPLD+microcontroller design flow.

XS95 Board Component Interconnections

The microcontroller and the CPLD on the XS95 Board are already connected together. These pre-existing connections save you the effort of having to wire them yourself, but they also impose limitations on how your microcontroller program and the CPLD hardware will interact. A high-level view of how the microcontroller, SRAM, and CPLD on the XS95 Board are connected is shown on the following pages. A more detailed schematic is also presented at the end of this manual.

The programmable oscillator output goes directly to a synchronous clock input of the CPLD. The CPLD uses this clock to generate a clock that it sends to the XTAL1 clock input of the microcontroller.

The microcontroller multiplexes the lower eight bits of a memory address with eight bits of data and outputs this on its P0 port. Both the SRAM data lines and the CPLD are connected to P0. The SRAM uses this connection to send and receive data to and from the microcontroller. The CPLD is programmed to latch the address output on P0 under control of the ALE signal and send the latched address bits to the lower eight address lines of the SRAM.

Meanwhile, the upper eight bits of the address are output on the P2 port of the microcontroller. The 32 Kbyte SRAM on the XS95 Board uses the lower seven of these address bits while the 128 KByte SRAM on the XS95+ Board gets all eight address bits. The CPLD also receives the upper eight address bits and decodes these along with the PSENB and read/write control line (from pin P3.6 of port P3) from the microcontroller to generate the CEB and OEB signals that enable the SRAM and its output drivers, respectively. Either of the CEB or OEB signals can be pulled high to disable the SRAM and prevent it from having any effect on the rest of the XS95 Board circuitry.

One of the outputs of the CPLD controls the reset line of the microcontroller. The microcontroller can be prevented from having any effect on the rest of the circuitry by forcing the RST pin high through the CPLD. (When RST is active, the microcontroller pins are weakly pulled high.)

Many of the I/O pins of ports P1 and P3 of the microcontroller connect to the CPLD and can be used for general-purpose I/O between the microcontroller and the CPLD. In addition to being general-purpose I/O, the P3 pins also have special functions such as serial transmitters, receivers, interrupt inputs, timer inputs, and external SRAM read/write control signals. If you aren't using a particular special function, then you can use the associated pin for general-purpose I/O between the microcontroller and the CPLD. In many cases, however, you will program the CPLD to make use of the special-purpose microcontroller pins. (For example, the CPLD could generate microcontroller interrupts.) If you want to drive the special-purpose pin from an external circuit, then the CPLD I/O pin connected to it must be tristated.

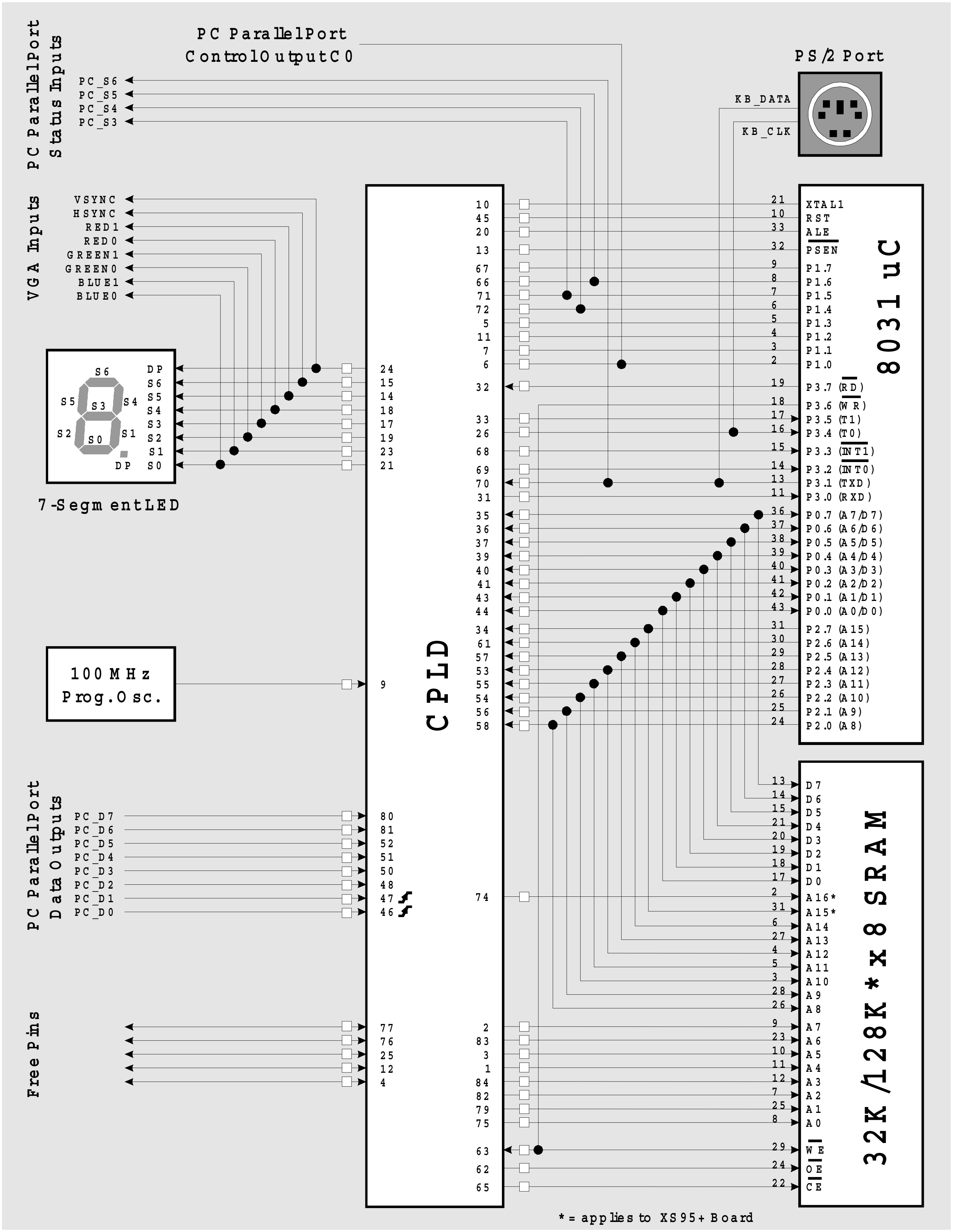
A seven-segment LED digit connects directly to the CPLD. (These same CPLD pins can also drive a VGA monitor.) The CPLD can be programmed so the microcontroller can control the LEDs either through P1 or P3 or by memory-mapping a latch for the LED into the memory space of the microcontroller.

The PC can transmit signals to the XS95 Board through the eight data output bits of the parallel port. The CPLD has direct access to these signals. The microcontroller can also access these signals if you program the CPLD to pass them onto the CPLD I/O pins connected to the microcontroller.

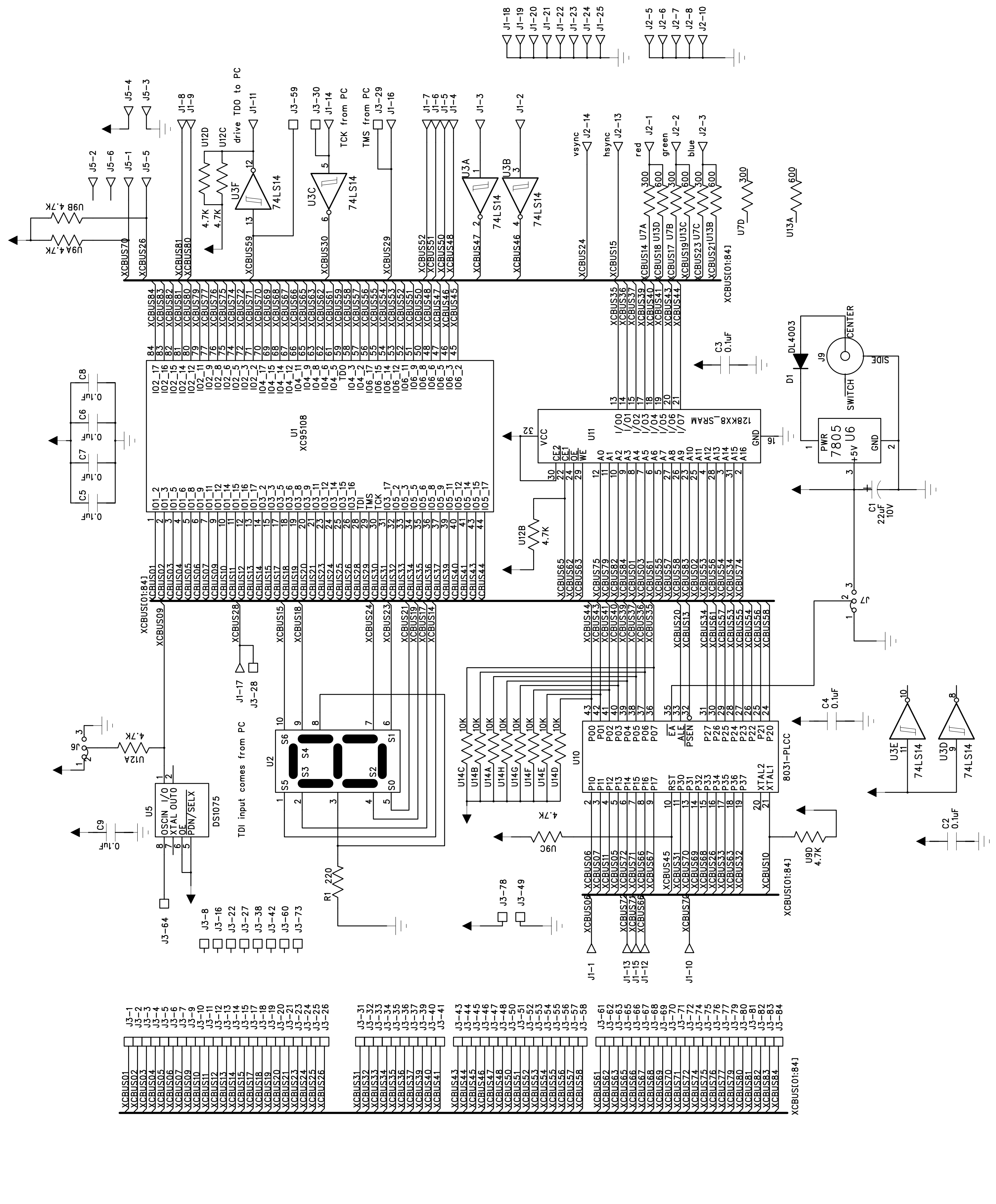
Communication from the XS95 Board back to the PC also occurs through the parallel port. The parallel port status pins are connected to pins of microcontroller ports P1 and P3 . Either the microcontroller or the CPLD can drive the status pins. The PC can read the status pins to fetch data from the XS95 Board.

The CPLD also has access to the clock and data lines of a keyboard or mouse attached to the PS/2 port of the board.

XS95 Pin	Connects to...	Description
21	S0,BLUE0	These pins drive the individual segments of the LED display (S0-S6 and DP). They also drive the color, horizontal, and vertical sync signals for a VGA monitor.
23	S1,BLUE1	
19	S2,GREEN0	
17	S3,GREEN1	
18	S4,RED0	
14	S5,RED1	
15	S6,HSYNCB	
24	DP,VSYNCB	
9	CLK	An input driven by the 100 MHz programmable oscillator.
46	PC_D0	These pins are driven by the data output pins of the PC parallel port. Clocking signals can only be reliably applied through pins 46 and 47 since these have additional hysterisis circuitry.
47	PC_D1	
48	PC_D2	
50	PC_D3	
51	PC_D4	
52	PC_D5	
81	PC_D6	
80	PC_D7	
10	XTAL1	Pin that drives the uC clock input
45	RST	Pin that drives the uC reset input
20	ALEB	Pin that monitors the uC address latch enable
13	PSENB	Pin that monitors the uC program store enable
6	P1.0, PC_C0	These pins connect to the pins of Port 1 of the uC. Some of the pins are also connected to the status input pins of the PC parallel port. The P1.0 port pin of the uC is also connected to the C0 control output from the parallel port.
7	P1.1	
11	P1.2	
5	P1.3	
72	P1.4,PC_S4	
71	P1.5,PC_S3	
66	P1.6,PC_S5	
67	P1.7	
31	P3.0(RXD)	These pins connect to the pins of Port 3 of the uC. The uC has specialized functions for each of the port pins indicated in parentheses. Pin 63 connects to the data write pin of the uC and the write-enable pin of the SRAM. Pins 26 and 70 connect to the clock and data lines of the PS/2 port. Pin 70 connects to a status input pin of the PC parallel port.
70	P3.1(TXD), PC_S6, KB_DATA	
69	P3.2(INTB0)	
68	P3.3(INTB1)	
26	P3.4(T0), KB_CLK	
33	P3.5(T1)	
63	P3.6(WRB), WEB	
32	P3.7(RDB)	
44	P0.0(AD0), D0	These pins connect to Port 0 of the uC which is also a multiplexed address/data port. These pins also connect to the data pins of the SRAM.
43	P0.1(AD1), D1	
41	P0.2(AD2), D2	
40	P0.3(AD3), D3	
39	P0.4(AD4), D4	
37	P0.5(AD5), D5	
36	P0.6(AD6), D6	
35	P0.7(AD7), D7	
58	P2.0(A8), A8	These pins connect to Port 2 of the uC which also outputs the upper address byte. These pins also connect to the upper address bits of the SRAM. Pins 34 and 74 are connected to the 128 KB SRAM address pins only on the XS95+ Board. Pins 34 and 74 do not connect to the 32 KB SRAM on the XS95 Board.
56	P2.0(A9), A9	
54	P2.0(A10), A10	
55	P2.0(A11), A11	
53	P2.0(A12), A12	
57	P2.0(A13), A13	
61	P2.0(A14), A14	
34	P2.0(A15),A15	
74	A16	
75	A0	These pins drive the 8 lower address bits of the SRAM.
79	A1	
82	A2	
84	A3	
1	A4	
3	A5	
83	A6	
2	A7	
62	OEB	Pin that drives the SRAM output enable.
65	CEB	Pin that drives the SRAM chip enable.
4	FREE0	These pins are not connected to other devices and can be used as general purpose I/O.
12	FREE1	
25	FREE2	
76	FREE3	
77	FREE4	


PC ParallelPort Status Inputs
PC ParallelPort Control Output C0
PC_S6
PC_S5
PC_S4
PC_S3
PS/2 Port
KB_DATA
KB_CLK
VGA Inputs
VSYNC
HSYNC
RED1
RED0
GREEN1
GREEN0
BLUE1
BLUE0
7-Segment LED
S6 S5 S4 S3 S2 S1 S0 DP
100 MHz Prog. Osc.
CPLD
8031 uC
XTAL1
RST
ALE
PSEN
P1.7
P1.6
P1.5
P1.4
P1.3
P1.2
P1.1
P1.0
P3.7 (RD)
P3.6 (WR)
P3.5 (T1)
P3.4 (T0)
P3.3 (INT1)
P3.2 (INT0)
P3.1 (TXD)
P3.0 (RXD)
P0.7 (A7/D7)
P0.6 (A6/D6)
P0.5 (A5/D5)
P0.4 (A4/D4)
P0.3 (A3/D3)
P0.2 (A2/D2)
P0.1 (A1/D1)
P0.0 (A0/D0)
P2.7 (A15)
P2.6 (A14)
P2.5 (A13)
P2.4 (A12)
P2.3 (A11)
P2.2 (A10)
P2.1 (A9)
P2.0 (A8)
32K/128K* x 8 SRAM
D7 D6 D5 D4 D3 D2 D1 D0
A16* A15* A14 A13 A12 A11 A10 A9 A8 A7 A6 A5 A4 A3 A2 A1 A0
WE OE CE
PC ParallelPort Data Outputs
PC_D7
PC_D6
PC_D5
PC_D4
PC_D3
PC_D2
PC_D1
PC_D0
Free Pins
* = applies to XS95+ Board

xs95v1_3.sch-1 - Tue Jul 27 00:52:18 1999



U1
XC95108
U11
128KX8_SRAM
U10
8031-PLCC
U6
7805
U5
DS1075
U2
74LS14
DL4003
drive TDO to PC
TCK from PC
TMS from PC
TDI input comes from PC
vsync
hsync
red
green
blue
XCBUS[01:84]