Abstract
Re-animating the PDP-11/70

©2004 John W. Peterson

When computers first appeared, one of their most distinctive things about them was the complex array of blinking lights and switches. For many years the rows of mysterious blinking lights were the very stereotype of what a computer was.

PDP-11/70 Front Panel

I’ve always missed the visual and tactile aspect of computers that’s been lost since the advent of microprocessors. When a console for a PDP-11/70 came up for sale on Ebay, I couldn't resist the opportunity to bring back to life the magic of those switches and lights.

This project uses the eZ80F91 as a network read controller to re-animate the PDP-11/70 panel. Every switch and light is available to the hardware and under software control. The eZ80F91 module is a perfect fit for this application.
While the eZ80F91 has a generous helping of parallel I/O, it clearly needs some more for this application. A Xilinx XC95108 CPLD is used to provide the additional parallel I/O. It’s very flexible and provides 69 I/O lines in a single package. Sixty-one of these lines interface to the panel and seven to interface to the eZ80F91. Another nice feature of the XC95108 is it interfaces directly to both the 3.3v logic of the eZ80F91 and the 5v logic of the panel.
The eZ80F91 takes on the rest of the panel interface, driving 9 status lines (via 10K pull-up resistors) and handling 13 inputs. The inputs from the panel (driven by TTL gates) are connected directly to the eZ80F91, since it is 5v tolerant. To drive the TTL gate inputs, the eZ80F91 GPIO ports are configured in “open drain mode” and pulled up to +5v via 10KΩ resistor networks. The remaining seven I/O lines are used to interface to the CPLD.

The I/O lines from the eZ80F91 and the CPLD are connected to the panel using the same ribbon connectors the panel originally used to connect to the PDP-11/70 CPU. These cables also supply some of the +5V power to the panel. The system is supplied with +5v from an external power supply. An LTC1086-3.3v regulator provides power for the eZ80F91 module. The only other components required are bypass capacitors, a reset switch, and connectors for programming the eZ80F91 and the CPLD.

Currently the software performs the basic, low-level access to the Panel. A library of routines (Panel_IO.c) interfaces with the GPIO ports to access all of the lights and switches. These are grouped by their placement on the panel, for
example the switch register, the address and data lights, the status lights and the various control switches.

The next phase of the software (currently under development) is to make the switch and light information available over the Ethernet. The Panel controller will listen for connections on a specified port, and report the switch settings and accept bit patterns for the light display. Higher-level operations (such as installing a particular animation pattern) are the next step. A simple sequence of bit patterns for display can be downloaded to the panel controller for playback.
Code Sample

// Project eZ2951 - Panel Controller for the PDP-11/70

#include <ez80.h>
#include <stdio.h>

#include "CPLD.h"
#include "Panel_IO.h"
#include "Lights.h"
#include "Timer.h"

extern void _init_default_vectors( void );

// This simple demo shows how to access the Panel I/O
// library.

void main( void )
{
    uint16 controls;
    uint8 speedNum;
    uint8 speeds[4] = { 100, 75, 50, 25 };

#ifndef ZSL_DEVINIT
    _init_default_vectors();
#endif

    InitTimer( 1 ); // 1ms interval
    InitZ80Ports();
    InitLights();

    /* main demonstration loop */
    while( 1 )
    {
        // Read the control switches
        controls = ReadControls();
        WriteDataLEDs( controls );

        // Spin the lights
        if (TestControlBit(controls, kSBusCycl))
            RevLights();
        else
            SpinLights();

        // Set the speed according to
        // the data knob setting
        speedNum = TestControlBit( controls, kDSel0 );
        speedNum |= TestControlBit( controls, kDSel1 ) << 1;

        wait( speeds[speedNum] ); // 50
    }
}