Homework #6

Out: November 9, 2022
Due: November 17, 2022 at 11:59 PM
Points: 130

CPLD Design

For this assignment, design and implement (with either a Lattice ispLSI1016 or a Xinlinx XC9572) the logic for **one** of the following two systems in VHDL. Both systems get data from an EPROM and output it either as video (to an NTSC monitor) or as audio (to a speaker using PWM). Both systems consist entirely of the Lattice/Xilinx CPLD, an EPROM, an oscillator, and some output circuitry. You will have to use the Lattice or Xilinx tools to design and implement the system. The CPLDs are in system programmable, so they can be programmed via the parallel port without removing the chip from the system.

Descriptions of both systems (including schematics and pinouts) are attached.

When you are done and have a working system you should demonstrate it to a TA and then hand in the VHDL source file along with any other supporting documents you may have. Be sure to fully document your design, including commenting the VHDL file. You must also submit your design and test vectors (VHDL file or files) electronically via the website.

PWM Audio Player (130 points)

System Description

The PWM Audio Player (Lattice version and Xilinx version) is designed to play any one of four messages stored in half of a 512Kx8 EPROM (27C040). There are four switches to select the message to be played and start playing it. Each message consists of data sampled at 8 KHz (thus there is approximately 32 seconds of messages in half of the EPROM). The messages are stored as indicated in the following table (the addresses and lengths are in hex, the times are in decimal):

Message	Switch	Start Address	Length
1	SW4	40000	8000 (4 seconds)
2	SW3	48000	10000 (8 seconds)
3	SW2	58000	24000 (18 seconds)
4	SW1	7C000	4000 (2 seconds)

The system works as follows. Each time a switch is pressed the message associated with that switch is played. The message should be played in its entirety. If the switch is still pressed when the message completes, the system should start playing that message again. Note that due to how the switches are defined to work there is no need to debounce them.

To play a message the CPLD must read each 8-bit sample from the EPROM (at the addresses indicated in the above table) and then output it at an 8 KHz rate. This means the EPROM address will change every 125 us.

Basic Design

To create an audio signal using PWM a pulse is generated whose width is proportional to the signal level desired. When this is averaged over time you get a reasonable reproduction of the actual analog signal. The CPLD is supplied with a 32 MHz clock so the output is 16x oversampled. Since there are 256 (8 bits) different analog levels to reproduce, a single sample takes 256 clocks. At an 8 KHz sample rate there are 4000 (assume 4096) clocks per sample. This leaves two choices, multiply the number from the EPROM by 16 and then generate a single pulse of that width, or generate 16 pulses, all of equal width for each sample. The latter gives much better audio output (more samples to average over) and will be used for this assignment.

Thus the system must read an 8-bit value from the EPROM every 125 us, and then output it sixteen (16) times using an 8-bit counter running at 32 MHz. To generate the PWM output, a one (1) is output when the counter is less than the value read from the EPROM and a zero (0) is output when it is greater than that value.

One complication that must be dealt with is that the access time of the EPROM is 120 ns (approximately 4 clocks). Thus there is not enough time to read the EPROM each output sample. The EPROM must be read and the data stored and then the new address is output all while still playing the previous data value (this is called pipelining). The first time the next audio data value is needed, the EPROM address will have been valid for almost 125 us, more than enough time for the data to be stabilized. That data can now be read and latched and again, the address incremented for the next value.

Extra Credit (40 points)

In addition to implementing the PWM output as described above, also implement it with no oversampling (effectively multiply the number from the EPROM by 16), and with no oversampling, but using an LFSR instead of a binary counter for the PWM. In order to easily compare the sound generated, pick a particular message to output and use the switches to determine which type of output to generate.

To implement the output with no oversampling just compare the data value with the upper eight bits of a 12-bit counter instead of the lower 8-bits.

NTSC Controller (150 points)

System Description

The NTSC controller (Lattice version and Xilinx version) is designed to output the color video image stored in a 512Kx8 EPROM (27C040) to an NTSC monitor. The image is stored with three bits for red and green and two bits for blue (this is not relevant since the non-CPLD hardware takes care of all of this). This image is stored line by line with each line consisting of 1024 pixels. Thus, the first line is stored starting at location 0 and the second line is stored starting at location 1024 (decimal). There are 480 lines (480 x 1024 -> 480 Kbytes). Note that there are only 512 pixels in the output so half of each line is not displayed. Thus less than half of the EPROM is actually used. The controller CPLD should continuously display the stored image on the monitor.

To display this image the controller CPLD must read it from the EPROM at the proper rate. It also must accurately generate all of the NTSC timing signals (SYNC and BLANK). These timing signals are described in greater detail below.

In the target system the controller chip runs at 10 MHz (instead of the standard 12.27 MHz). Because of this, some of the timing numbers given differ slightly from standard NTSC values. There are two major types of timing to consider, horizontal timing describes the timing for an individual line and frame timing describes the timing for an entire screen. For a clearer understanding of this timing refer to the attached timing diagram while reading the descriptions that follow.

NTSC Composite Video Signal Format

Within a frame (screen) on an interlaced display, there are two fields, the even field and the odd field. The even field is output first and consists of the even video lines (stored as lines 0 to 239 in the EPROM). The second field output is the odd field and consists of the odd video lines (stored as lines 256 to 495 in the EPROM). The monitor has 525 actual lines of output, but 45 of these are blanked (no signal is output) and are used for vertical synchronization (leaving 480 lines of video). The blanked lines are lines 1-20, lines 261-283, and lines 524-525. Within each field, the first nine (9) lines (lines 1-9 and 263-272) are used for vertical synchronization. These lines consist of three (3) lines of equalizing pulses, followed by three (3) lines of serration pulses that do the actual vertical synchronization, then followed by three (3) more lines of equalizing pulses. Since this starts vertical retrace the next eleven (11) lines of data are also blanked (as mentioned above). This means line 21 contains video line 0 (first line of the even field at ROM address 0x00000) and line 284 contains video line 1 (first line of the odd field at ROM address 0x40000).

Since the number of lines (525) does not divide evenly by two, each field actually contains 262.5 lines. The monitor also uses the half-line information to distinguish between even and odd fields. This means the positioning of some pulses within a line of an odd field is different from that of an even field.

There are four (4) different types of lines that are generated for NTSC video: lines containing equalizing pulses, serration pulses, blanked video, and normal video. Equalizing pulses are output on lines 1, 2, 3, 7, 8, 9, 264, 265, 270, 271, and half of 263, 266, 269, and 272. Serration pulses are output on lines 4, 5, 6, 267, 268, and half of 266 and 269. Blanked video is output for lines 10 to 20, lines 261 to 263, lines 273 to 283, and lines 524 to 525. The rest of the lines (21 to 260 and 284 to 523) are normal video. The timing for each type of line is given in the attached timing diagrams. Note that all of the lines start with the output at the "sync level" and that each line is of the same length (63.6 us). Some lines (equalization and serration) contain two, evenly spaced, "sync" pulses, while the other lines (blanked or normal video) contain only one "sync" pulse.

As indicated on the attached timing diagrams there are three "levels" to be output. The lowest level is the "sync level" and is output by forcing the SYNC line high and the BLANK line low. The middle level is the "blank level" and is output by forcing both the BLANK and SYNC lines low. The "data level" is output based on the data from the EPROM and is output whenever the BLANK line is high and the SYNC line is low. (Note that the BLANK signal is active low and the SYNC signal is active high from the perspective of the CPLD.)

The timing of the SYNC and BLANK signals is given in the attached timing diagrams (both their position within a line and their width). For most types of lines the BLANK signal is always active (low). Only for a normal video line does it go inactive (high). Within a normal video line the sync pulse and back porch (first 10.9 us) are blanked. The front porch at the end of the line (last 1.5 us) is also blanked.

Basic Design

The system consists of a number of counters to keep track of the current line number on the display and in the EPROM. These counters are used to run a state machine and to form the EPROM address.

One complication in this system is that a new pixel is output every 100 ns, however the EPROM access time is 90 ns. If it were connected directly to the video output circuitry, the display would be full of garbage since only in the last 10 ns of each pixel would the real data be present. Instead the EPROM must be read ahead of time (pipelined) and latched (using the 74273). This means the output EPROM address is always one more than the pixel address.

Warning: You should **not** hook your system up to an actual monitor until the monitor signals have been checked by a TA using an oscilloscope. This is so that the monitor will not be blown out by design errors as you are debugging your design.

Extra Credit (40 points)

A second EPROM is available which contains 32 frames of 128x128 interlaced data. Thus these could be played sequentially and, at full speed, one full second of video would be output. For the extra credit, implement the NTSC controller to display these 32 frames (at any speed you want). You will need to place the 128x128 picture somewhere on the screen and the remainder of the screen should be blanked.

Homework #6 Resources

- PWM Audio Player schematic (Lattice version and Xilinx version)
- NTSC controller schematic (Lattice version and Xilinx version)
- NTSC Timing Diagram
- Homework Q&A
- Electronic Submission

Last updated September 27, 2022 09:17 AM by glen@caltech.edu