CpE311 – Semester Project

Fall Semester, 2002.
DUE: Thursday, Dec. 5, in class.

For your project, you will design an A/D converter with a full 8051-microcontroller interface. The project will be designed for implementation in an Actel 40MX02 FPGA. The project may be implemented either using VHDL (strongly recommended) or using Design Architect and schematic capture. Details are given below.

Background:

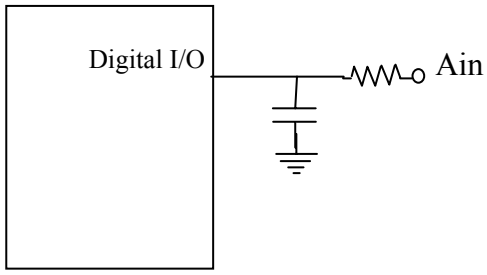

Fig. 1

A crude but effective A/D converter can be constructed from a digital I/O port using an external resistor and capacitor (Fig. 1). The analog voltage is calculated from the time it takes the analog voltage input to charge the digital I/O to a 1 or a 0. The procedure is to:

1. Write a 0 (or 1) to the digital I/O port to discharge (charge) the capacitor
2. Configure I/O port for input – let cap charge (discharge)
3. Count how long it takes for cap to charge (discharge) to a 1 (0):

```
count = 0;
While (Digital I/O input is 0 (or 1)){
        count = count + 1;
}
```

4. scale count appropriately to calculate Ain.

The amount of time it takes to charge to a 1 (discharge to a 0) depends on the equation:

$$V_{I/O} = V_{in} + (V_0 - V_{in})e^{-t/RC}$$

where V_0 is the initial value of the capacitor (either a high – around V_{DD} – or low – around V_{SS}). In particular, you are looking to find the time it takes to charge from a V_{DD} (or V_{SS}) to a logical 1 (or 0). The value of V_{IH} or V_{IL} can be found from the 40MX02 datasheets at http://web.umr.edu/~daryl/classes/cpe311/40mx.pdf.

Your design will be built to interface with the 8051. I will give a handout in class that shows how the interface works. In summary, the 8051 accesses external RAM by:

1. Writing the low byte of the address to be accessed out port 0 and the high byte of the address out port 2. Port 0 is a multiplexed address and data line, so the low address byte will have to be latched externally.

2. Changing ALE – address latch enable – from high to low, allowing the low byte of the address to be latched externally.
3. If reading from RAM, making Port 0 go into a "high impedance" state so that RAM has control of the data bus. If writing to RAM, putting the data to be written on Port 0.
4. If reading, making the RD/ line go low telling RAM to write the data back to the 8051. If writing, making the WR/ line go low telling RAM to latch onto the data on the data lines.
5. Going on to the next read/write cycle. The new cycle begins with RD/, WR/ and ALE high and a new address written out on Port 0 and 2.

Specification:

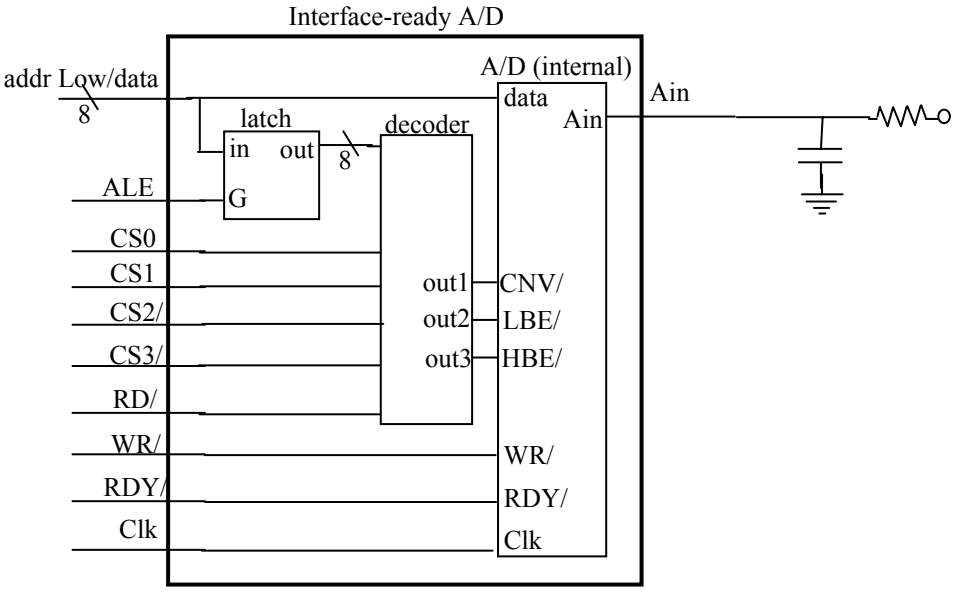

Fig. 2. Block diagram of specified project.

Your chip should be implemented at two "levels" as shown in Fig. 2. The outer level is ready to interface to the 8051. The inner level contains a address latch, an address decoder, and your A/D design. Specifications include:

- **Precision and accuracy.** I will let you specify the precision and accuracy of your design. The more the better, but you'll have to convince me that your solution is correct. Use information from the 40MX02 datasheet to back up your contention. To keep consistency between groups, I would like everyone to design the interface as a 16-bit interface. If your design has less than a 16-bit precision, then write 0s in the upper byte (for example, if you designed an 8-bit converter, the largest value your converter would write back would be 0x00FF).
- **Function.** The minimal-featured (internal) A/D converter for this project should be able to perform four functions:
    - A/D conversion. When the CNV/ (convert) line goes low, the A/D converter will begin to convert the analog signal on its Ain line to a digital value.
    - Ready. When the conversion is complete, it will set its RDY/ low.

- Read Low Byte. When the LBE/ line goes low, the A/D converter will write the low byte of the converted analog value to the data lines. The data lines will go back to "high impedance" when LBE/ goes high again.
- Read High Byte. When the HBE/ line goes low, the A/D converter will write the high byte of the converted analog value to the data lines. The data lines will go back to "high impedance" when HBE/ goes high again.

An A/D converter with similar function can be found at http://www.analog.com/productSelection/pdf/ad573.pdf.

- **Interface.** The general interface for your design is shown in Fig. 2. Your A/D converter will be designed with an internal address latch and address decoder, allowing easy interface to the 8051. The decoder will be designed such that the A/D converter is "selected" when CS0=CS1=1 and CS2/=CS3/=0 and one of the following (low-byte) addresses is accessed:
  - 42H: When reading from this address, CNV/ will go low.
  - 43H: When reading from this address, LBE/ will go low.
  - 44H: When reading from this address, HBE/ will go low.

  The testbench I give out later this semester should help some with the interface.
- **Size.** Your design should be as small as possible. It should fit into an Actel 40MX02.
- **External R and C.** You may specify. I suspect you will find a tradeoff here between speed and accuracy.
- **Clk.** You may specify the speed of the clock. The clock is necessary to complete the design.

Testing: A short testbench will be provided for you later in the semester. At a minimum, your design should pass this testbench; however, you are also responsible for more complete testing of your project. Your project will be tested by another much more complete testbench of my own after it is turned in.

Teams: Projects will be done in teams of 2 of your own choosing. Team member contribution will be evaluated in the final report and through a confidential evaluation the last day of the semester. Please do not share your team's solution with other teams.

Extra Features: You may improve your design by adding extra features of your choosing, however you may not change the fundamental spec. I have provided a WR/ in the basic specification to give you the option to write data to the A/D as well as to read, for example to write calibration data to the A/D. You may provide additional lines from the decoder to the A/D or add additional parts inside the overall design. If you wish to add additional lines to the "outside world", ask for permission first. You may use a different scheme for A/D conversion too, but please ask for permission first there too, as it may invalidate the testbench. In general, I would not like any changes that will invalidate my testbench.

Report: Your completed project will be documented in a final written report. The report itself (not including source code, schematics, or simulation results) should not exceed 10 pages unless you get special permission from the instructor. *At a minimum*, your report should include:

- Title and team members
- Summary. Should summarize entire project in 200 words or less. Be as specific as possible (similar to cliff notes).
- An explanation of your design rationale
- An explanation of your test rationale, including testbenchs, results, a logical explanation of why your test results/methods convince you your design REALLY works (think of me as your boss, getting ready to spend $100k getting your design fab'd. I wouldn't be happy if I spent money to fab it and it didn't work. Convince me to spend the $$). Testing with my testbench is the minimum requirement.
- If your design did not work, give me an explanation of why it didn't work and what would be needed to fix it.
- Work effort distribution. List each person in your group. Tell what their job was and the total percentage effort they contributed to the completion of the project.
- Design "datasheet". The datasheet will summarize the design's features and performance specifications. For example, conversion time, size of R and C used, precision, accuracy, range, etc. A very good example of a word-based datasheet is given at http://www.umr.edu/~daryl/classes/cpe311/ds1822-datasheet_v2.doc. (Yours does not have to be so complete)
- Hardcopy of well-commented source code or schematics, included in the appendix.
- Electronic copy of project. You may send me the electronic copy via email. Your entire project should be tar'd and zipped into a single file. To understand how to do this, do a "man" on tar and gzip on a Unix machine… I believe the following command should work: "tar –cf – *directory_name* | gzip -9 > yourname.tar.gz", where *directory_name* is the name of the directory containing your design and yourname is YOUR name (so I can easily identify which files belong to whom). This command should be issued from the directory containing your design directory (the parent).

Grading: Grades will be given according to several criteria. Criteria will include the information given in the report, the form and style of your report, performance of your design, especially with my own testbench, timeliness in meeting deadlines, design features – especially "extras" –, project size, speed, and accuracy, and design creativity and elegance.

Plagiarism: I would not be surprised if you find similar designs out on the web or from past CpE311 groups. You are responsible for building your own design. If you use ideas from someone else's design, you must clearly reference their design in both your report and in your VHDL code/schematic. Failure to do so will bear very unpleasant consequences. If you have any doubts about using someone else's ideas, please ask!