

FAQ

(Last FAQ added October 23, 2006 on Page 5.)

Self-assessment Assignment 1: 1. Can we do this in groups?

Answer: Ideally done individually. If you do it in groups, I allow it only under the following condition:

you take the responsibility to make sure that you *individually* understand the solution *as if you did it all by yourselves*.

2. What is the “add function” talking about? How about the adder/subtractor from the component library?

Answer: There are two parts to this assignment.

- The first part is to solve your problem using VHDL code (all HDL and no schematics). In that part, if you use a counter to represent the numbers 0 through 9 and perform addition on this counter, you may use the VHDL “+” operator to effect the incrementing.

For those not using such an adder in this part of the assignment, they can ignore the “(including the add function)”. I know that some of you are solving this part using a straight 7-bit representation for the state. The advantage of such an approach is that you can then feed the state bits directly to the LEDs!!

- The second part is to solve the problem using schematics. Here I require that you use a counter and an adder, and add a “1” to the counter to take it to successively higher values. Then use a library unit such as AddSub16 and construct a schematic around it. The rest of the design is up to you to detail.

UCF File Creation: Select your .vhd file in the sources window. In the process view window you should see User Constraints. Open it up, double click on Assign Package Pins (to use the Xilinx PACE tool) or Edit Constraints (if you want to edit the UCF by hand).

Using the PACE tool:

If the pin list isn't visible when it starts, use the "View -> Design Object List" menuitem. You should see all your connections listed. In the "Loc" column, you can type in your pins ("p26" for S4, for example).

--

Neal

Password protected webpage creation:

Subject: password protected webpage
From: agreer@eng.utah.edu
Date: Fri, 25 Aug 2006 20:03:54 -0600
To: cs3710@cs.utah.edu

There have been some questions about how to password protect your project webpage. It's not very hard to do, but some attention to detail is required to make it work.

You need to create two special files: `.htaccess`, and `.htpasswd`
Some information about how to create these files can be found here:
<http://www.freewebmasterhelp.com/tutorials/htaccess/3>
http://www.kxs.net/support/htaccess_pw.html

The permissions need to be set to read and exec for both files for you, other users

I hope this helps some.

-Aaron

FLASH programing: See <http://www.cs.utah.edu/classes/cs3700/lectures-s03/lec14/flash-programming.txt>

CPU manual updates - credits given where I can recall - let me know if any omissions:

Updates:

- Made `sex()` and `zex()` operate always on 8-bit quantities, yielding 16-bit quantities
- Made `PC + 0x0001` a standard PC update idiom (the former method was not clear)
- Simplified (and made uniform) `lsh` and `lshi`
- Other changes to tidy up the document were made (some aspects of the previous document were simply wrong – largely when it came to examples – or misleading)
- The manner in which the assembler encodes things is not yet defined. That will be made clear in another description. `cpummanual` only tried to convey what the machine does
- Typo for `mov` – must move from `Rsrc` – fixed it
- Our CPU does not have a “halt” or “stop.” Most CPUs will spin in an idle loop till being interrupted out of it. The same method will be followed for our MCR16 versions. For your list of features to add to MCR16-ex, I strongly suggest adding interrupts. This will give new capabilities for your CPU (the ability to respond to external interrupts through interrupt handlers) without being too hard to implement.

Assembler manual updates: These changes were made on 9/13/06:

- I fixed some typos – “13” (el three) changed to “13” (thirteen)
- I fixed opcode for `movi` – must be a D not a B
- I changed displacement for “`bcs L6`” – has displacement 7 not 6
- I allowed you to implement `r8`, `r9`, `rA`, `rB`, `rC`, `rD`, `rE`, and `rF` (to obtain 16 regs) OR you can use these “extra” registers to trigger graphics-space writes. Required to support only 8 registers `r0` thru `r7`.

Assignments due next week - also meetings on Tue and Thu: Hi

>Hi,
>

>This is the assignment description due on Sep. 19.

>

> " 1. The group members must show me how they are proceeding with
>the construction of the assembler and the simulator. I will check your
>understanding at least as far as the thin-slice instructions are
>concerned. All the documentation must be there on the webpage also.

>

> 2. We will have given you a dual-port unit with VGA and mock CPU.
>You must show evidence that you have used this design, and show the
>extent to which your circuit works."

>

>I'd like to make clear what are the requirement.

>

>For 1, what I know are:

>The assembler is that it generates the machine code in the listing
>format and hex format. I can see the specification from the handout.
>But I don't know how to make it.

Fine - the idea is to motivate you to think hard and ask us questions! We are not asking

> Also about the simulator, I think I

>didn't learn at all. I don't know how to show you how to construct
>the assembler and simulator. Could you give me an advice?

You have to think thru writing the simulator, see where you get stuck
and ask us for help. This class is one where people *won't* be taught
all details -- but they will be given enough instruction (like I've
given so far) and then asked to try doing it, get stuck, and ask us.
This is the only real way to learn large-scale SW / HW construction
which this class is about.

So I want you to ask questions on writing a simulator (a computer
program that simulates the MCR-16 instructions and prints regs / mem
before/after each instruction). This ability is expected.

Don't worry before asking questions ("stupid questions" are also fine
to ask). Starting early is crucial -- stupid questions asked late
won't be welcome.

>

>For 2, you expect me to do demo of VGA and mock CPU as TA did on Tuesday?

No, report on your experience. We will trust your words.

>

>Also the due day of the assignment that said:

>

```
>"You must present evidence that you can successfully use the Block RAM
>memory. I will be assigning a simple design (Fibonacci circuit) that
>you must show working (evidence on webpage)."
>
>is next Tuesday? Or next next Tuesday, that is Sep. 26?
```

I had assigned two things:

- play with fibonacci assigned to you (run it and see that it works -- takes 5 minutes)

Hence I want this shown next Tue

- develop the "largest" circuit that was explained today in class. This requirement wa

I'd like to see this circuit also work on Tue. If you can't do it, ask us questions d

I won't lecture next week on Thu -- because I did not get to meet the even numbered groups. So next week, on Tue, I'll meet the odd numbered groups and on Thu meet the even numbered groups. During this time, everyone should demo the "largest" circuit.

-- end

- FAQs during the labs of 10/3 and 10/10:**
- **Stereo codecs:** Search term in www.xess.com gives you application note leading to all the files you need for a loop-back test (feed analog input; it includes circuitry to emit back same analog signal after a/d and d/a). Should be able to figure out a/d using this app note.
 - **The MC68000 instruction set:** Google searching this term gives you the details for MC 68000's instruction set (was included in one of the project suggestions).
 - **PS2 data sheets** - should also be there on the xess site.
 - **Placement:** There is a "view/edit placed design" menu which will show you how your mapped design is going to reside on the FPGA. You'll also find the "mapping report" useful to see (how much of the FPGA have I consumed?)
 - **Points for re-using modules at xess.com:** Please look at the project selections webpage for other general instructions.
 - **Mapping error:** When you synthesize and map designs for next week's checkoff (Oct 17th HW thin-slice), you may find that ISE does not like switches such as S5 being used as a clock. One solution is as follows:

```
myIBUF : IBUF(I => S5, 0 => S5_int);
```

Then use S5_int as your "internal" S5 signal.

One typo in the assembler manual: Check the symbol table value for L6 in the assembler manual given out during Lecture 3 (3:9-7). It should have been 080A, not 0809.