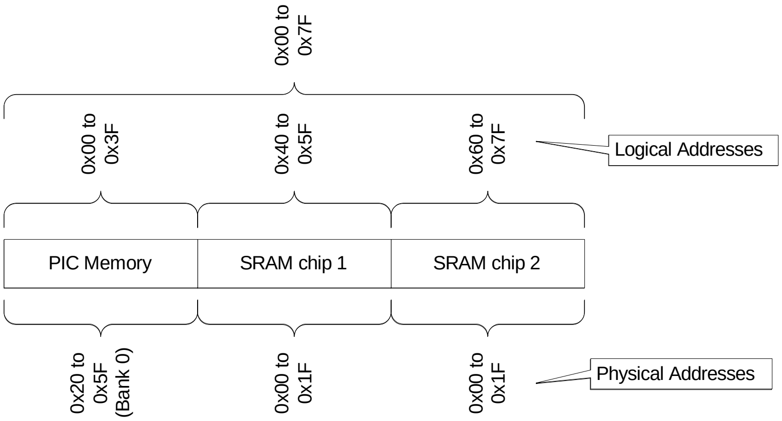
ECE 354 – Lab 3: Connecting the PIC to SRAM

In this lab, the PIC will be connected to external memory (SRAM). You will use simplified version of the PIC-PLD bus from Lab 2 to communicate with the PLD, which in turn manages the control signals for the SRAM. For the demo, you need to write a memory test program that can verify the correct operation of any range of memory locations in the virtual address space. The physical memory consists of PIC-internal memory and two external SRAM chips (see Figure). The user should be able to specify via the terminal the range of addresses to be tested and the test pattern to be used (i.e., the value that is written to memory). The PIC writes the pattern to all memory locations in that range, then reads the memory values, and verifies if the values are equal. The results of the test should be displayed on the terminal (either success or failure).



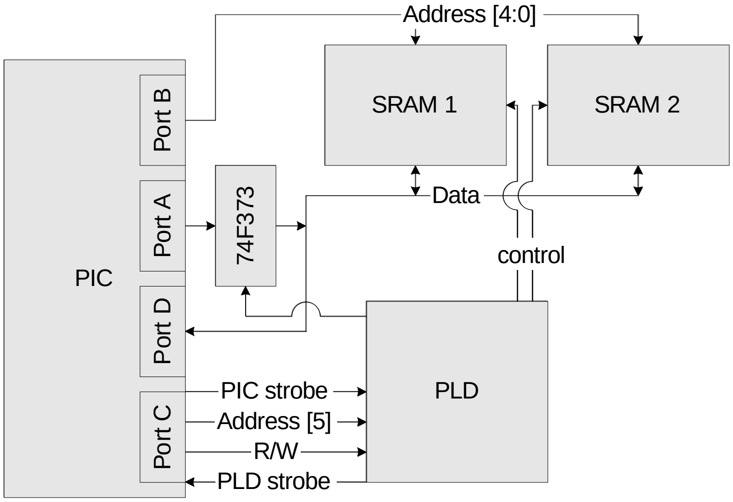
Tasks

For this lab you must complete the following tasks.

1. This VT220 interface should query the user for the start address of memory test (“Enter start address [0x00..0x7f]:”), the end address (“enter end address [0x00..0x7f]:”), and the 8-bit test pattern (“enter test pattern [0x00..0xff]:”). You may use polling (easier) or interrupts (for those who don’t find enough challenges in this project already) for reading from the serial port. It is required that you use the “advanced string printing” method with the RETLW instruction discussed in class. Your program must check the input for correctness (i.e., start address ≤ end address). If incorrect values are entered, you may just jump to the beginning of the input routine and start over (no error message required).
2. The memory test will work as follows: You will test each memory location in the range with eight patterns. These patterns are the input pattern shifted left 0-7 times. You should first write the first pattern to all memory locations within the range, then read them and verify correctness (do not write, read, and check them one-by-one). Then repeat with the second through eighth pattern.
   Example: range 0x5f..0x60, pattern: 0x21. The eight patterns for testing are 0x21, 0x42, 0x84, 0x09, 0x12, 0x24, 0x48, 0x90. The sequence of operations is [first pattern] write 0x21 to address 0x5f, write 0x21 to address 0x60, read and verify address 0x5f, read and verify address 0x60, [second pattern] write 0x42 to address 0x5f, write 0x42 to address 0x60, read and verify address 0x5f, …
   The results of the memory test should be displayed on the terminal. If the test is entirely successful, it should say so. If there is an error (i.e., read value does not correspond to written value), each error should be reported (e.g., “test error at address 0x5f (wrote: 0x21, read: 0x20)”. After the test, the program should allow the user to run another test, etc. The PIC should operate in stand-alone mode.
3. For testing internal PIC memory (virtual address 0x00 to 0x3f), you should use indirect addressing as discussed in class.
4. For the interface to the two HM6264B SRAM chips, you should set up the PLD to handle the control signals for the SRAM. The address and data values are transferred directly (i.e.,

not via the PLD) between PIC and SRAM (see Figure), except for addr[5], which is used to select between the SRAM chips (partial address decoding). The output port A of the PIC is isolated from the data bus via a 74F373 tri-state buffer. Since we use only a small part of the SRAM, it will be necessary to connect the upper nine address bits of each chip directly to ground. A state machine should be created for the PLD that not only coordinates handshaking but also allows for the correct generation of SRAM chip select (CS), write enable (WE), and output enable (OE) signals as well as an output enable (OE) for the 74F373 buffer. Note that only one OE signal should ever be active at a given time.

5. To test your system, make sure the test runs without errors. Then introduce errors by disconnecting data wires (or pulling them to ground). You will probably not be able to create errors by disconnecting address wires (why?).



Demo

During the demo, your PIC should perform the memory test robustly for any range of addresses and test pattern. I will test the correctness by pulling various address and data wires from the SRAM to verify that the test fails and prints the expected failure messages. You should also set up your lab to show the SRAM transactions on a logic analyzer (or show printouts). You do not need to show the PIC–PLD interactions.

Getting Started

One possible approach to implementing this lab is the following:

1. The PIC–PLD address and data transfers from lab 2 should be modified to target the SRAMs rather than the PLD. Note that the number of address bits is now five rather than four.
2. After coding up and simulating a state machine to drive the SRAM control signals, take a look at the signals on the waveform analyzer. Do they look like the signals in the SRAM data sheet and class handout?
3. Wire up the SRAMs to your new interface. Establish communication with the terminal. Note that following initialization the 16F877 should poll for data receipt at the UART similar to Lab 1.
4. Rather than trying to debug the entire SRAM interface all at once, try to read and write a value to a specific RAM location (e.g. physical address 0x00).
5. Once you have verified the hardware is working correctly, the rest of lab should just be 16F877 coding.

Lab Report

The lab report guidelines are the same as for previous labs. Your report should include a schematics and a logic analyzer printout of SRAM read and write transactions. It is not required show PIC–PLD transactions.