8051 based Embedded System

EE4380 Fall 2001

Pari vallal Kannan
Center for Integrated Circuits and Systems
University of Texas at Dallas

8051 as the System Controller

- Requirements
  - CODE: Some way of storing the code and make the 8051 execute the code
  - RAM: Some memory for variables, stack etc
  - Application dependant Peripherals
- Choice of all these components is dependant on the particular flavor of 8051 used

Program CODE

- On chip (E/EE/NV/Flash) – ROM
  - If not present or not enough, then ***ADD*** external ROM
- How to get the code onto the ROM ?
  - PROM programmer
  - Swap prom chips or 8051 chips
  - In system programming
    - Download code onto programmable ROM (usually Flash/NV)
- Totally do away with ROM.
  - Download code directly onto RAM and use RAM as code memory
  - Valid option is another system is available
  - Common on development boards

PROM Comparison

Type	Volatile ?	Writeable?	Erase size	Erase Cycles	Cost /byte	Speed
SRAM	Yes	Yes	Byte	Unlimited	Expensive	Fast
DRAM	Yes	Yes	Byte	Unlimited	Moderate	Moderate
Masked ROM	No	No	N/a	N/a	Cheap	Fast
PROM	No	Once*	N/a	N/a	Moderate	Fast
EPROM	No	Yes*	Full chip+	Limited	Moderate	Fast
EEPROM	No	Yes	Byte	Limited	Expensive	Fast read, slow write
Flash	No	Yes	Sector	Limited	Moderate	Fast read, slow write
NVRAM (SRAM with backup)	No	Yes	Byte	Unlimited	Expensive	Fast

- * need programmer to write
- + need UV Light to erase
- From http://www.embedded.com/story/OEG20010416S0068

Common Devices

- UV erasable EPROM
  - Approx street price = $5 for 256-15 part
  - 27C128, 27C256, 27C512-15
  - C – CMOS, 128 = 128 kilo bits = 128/8 = 16Kilo bytes, organized as 16Kx8
  - Last two digits is speed grade
    - 15 = 150ns access time
- EEPROMs
  - 28C64, 28C256-15
  - Approx street price = $10 for 256-15 part
- Flash
  - 28F256, 28F010 (1024 kbits, 128Kx8)
  - Approx street price = $7 for 010-15 part
- SRAM
  - 6264, 62256-10
  - Approx street price = $4 for 256-12 part

8051 Memory Interfacing

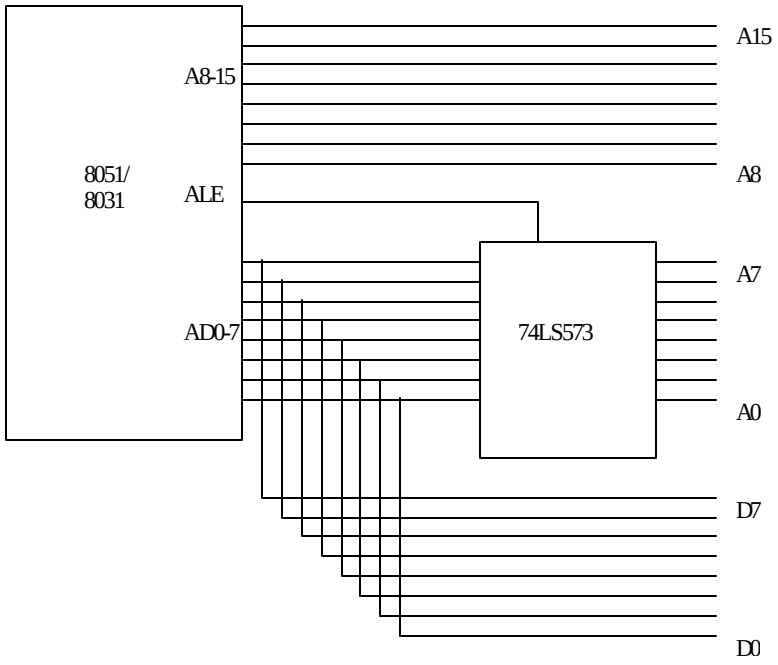
1. Decide the memory map
   - System and application dependant
   - Simple solution is to carve out the 64k of data memory and 64k of code memory
   - To address more than 64k use additional bits from unused I/O ports as address lines
2. Decide on components
3. Use a decoder on the address lines to generate CE signals for the devices
4. Use PSEN for code memory
5. Use RD, WR lines for data memory
6. EA = V_{cc} for on chip code rom, EA = GND for external code ROM access.

Basics of Interfacing

- Memory (and almost all peripherals) have special control lines for microprocessor interfacing
  - CS, CE : chip select or chip enable
    - Driven by the address decoder from the processor
    - Usually active low
    - When asserted, the chip/peripheral is selected
  - OE : output enable
    - Commonly seen on memories
    - Turns on the output tri-state buffer inside the memory
    - Sometimes referred to as RD in Read-write (SRAM) memories
  - WR : write enable
    - Driven by WR of the processor

8051 Specifics – DEMUX AD0:7

- Use a 8bit latch 74LS573 /373

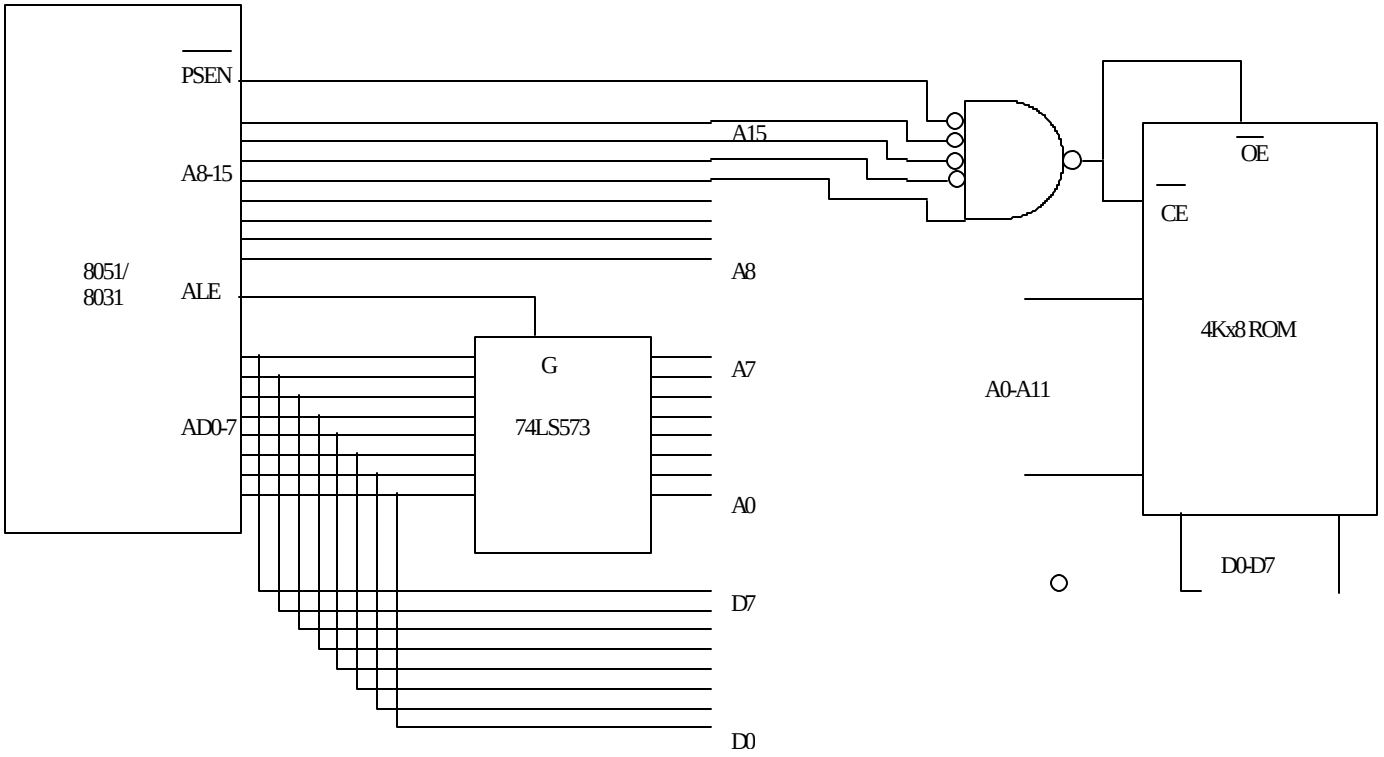


External Code ROM 4K at 1000H

- 1000H = 0001 0000 0000 0000b
- Address lines of interest
  - A15, A14, A13, A12
- Objective
  - Design a logic circuit to generate an active low signal when [A15:A12] = 0001 and PSEN=0
- Address map is 1000H to 1FFFH
  - No repetitions
  - If any of A15:A13 not used in the decode then the same memory block will repeat at different addresses

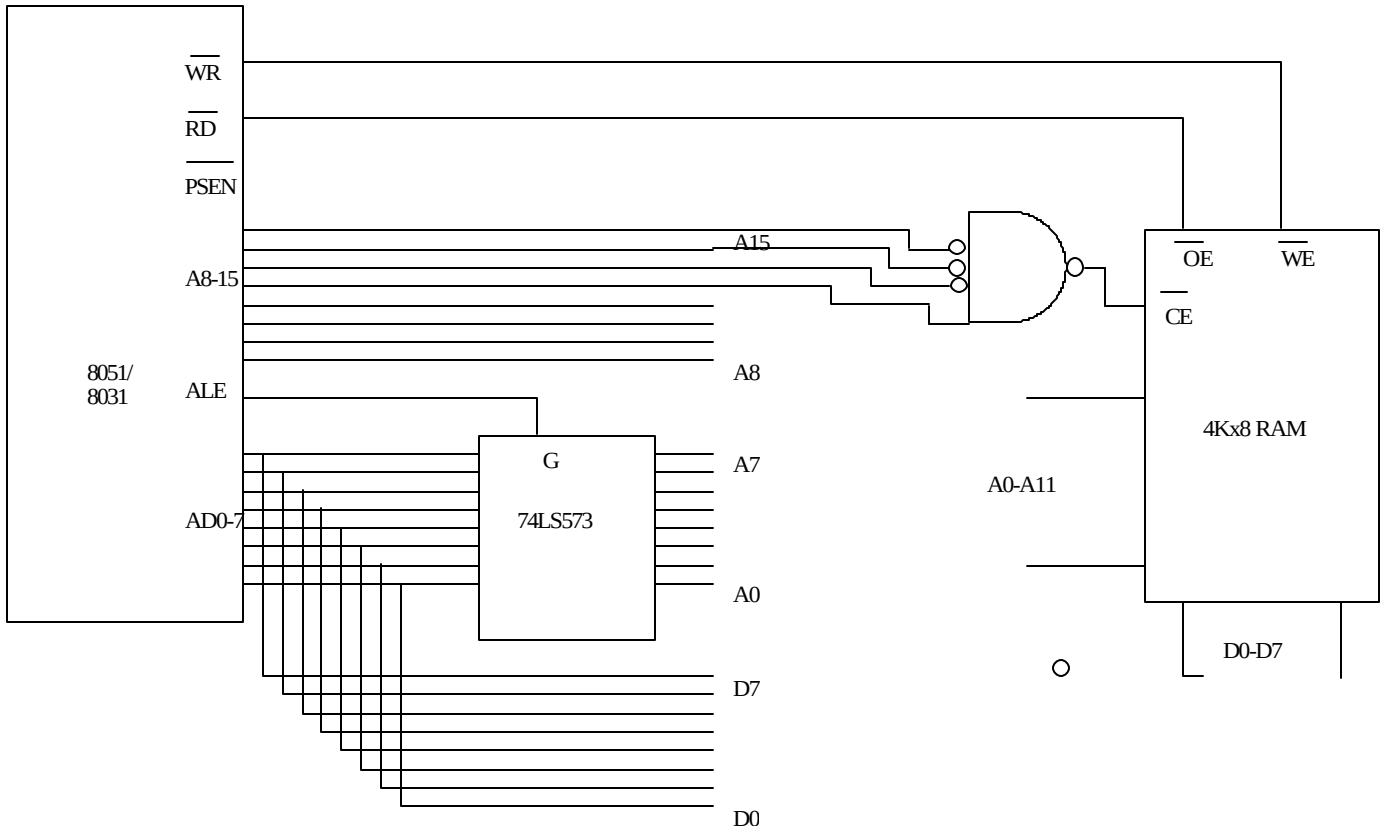
External Code Rom (contd.)

- PSEN has to be used for code memory



External Data RAM at 1000H

- PSEN is not used. RD and WR are used



Board Design

- Photo-fabrication process
- Wire wrap
- Bread board