Org: x86 Memory and I/O

Alex Chi

Update: April 14, 2020

Contents

1 Objectives **2**

2 Memory Address Decoding **2**

2.1 NAND 2

2.2 74LS138 3

2.3 More methods 3

2.4 Addressing 3

2.5 Checksum 3

3 Memory Organization in 8086 **3**

4 I/O **4**

4.0.1 Insturctions 4

4.0.2 Direct I/O 4

4.0.3 Indirect I/O 4

4.0.4 16-bit 4

4.0.5 Interfacing 8-bit I/O to 16bit bus 4

4.0.6 Output Port Design 4

4.0.7 Input Port Design 5

5 Example: 8255 5
5.1 Packaging 5
5.2 Internal Structure 5
5.3 Operation Modes 5
5.3.1 Mode 0 6
5.3.2 Mode 1 6
5.3.3 Mode 2 7
6 Example: Timer 7
6.1 Internal Structure 8
6.2 Mode 0 8
6.3 5 Modes 8

1 Objectives

- how does 8086 execute `MOV BX, [1000h]`
- m-r or r-m?
- what's different between I/O or memory

2 Memory Address Decoding

- CPU calculates physical address
- memory address decoding circuitry

2.1 NAND

- A0-A14 to addr
- A15-A19 select 10000, nand to cs

2.2 74LS138

- ground G2A, G2B
- G1 = 1
- HI:LO

2.3 More methods

- absolute address decoding (use all lines)
- linear select decoding (only selected lines are decoded)

2.4 Addressing

- linear
- absolute

2.5 Checksum

- checksum byte: add all bytes and take 2's complement
- parity bit: even / odd parity
- 8086: odd parity (odd numbers of 1 - parity set)

3 Memory Organization in 8086

- refer to figures in slides
- use odd and even banks
- even bank CE - A0
- odd bank CE - BHE
- even bank to D0-D7
- odd bank to D8-D15
- `MOV AX, [1000H]` A0=0, BHE=0
- `MOV AH, [1000H]` A0=0, BHE=1
- `MOV AX, [1001H]` A0=1, BHE=0; PA'=PA+1, A0=0, BHE=1
- unaligned access requires 2 bus cycles

4 I/O

- isolated I/O
- special instruction

4.0.1 Insturctions

- IN dest, source
- OUT dest, source

4.0.2 Direct I/O

- IN AL, port# (0-ff, only AL)
- OUT port, AL

4.0.3 Indirect I/O

- port# 0-ffff
- only DX
- MOV DX, port#
- IN AL, DX
- OUT DX, AL

4.0.4 16-bit

- AL replaced by AX

4.0.5 Interfacing 8-bit I/O to 16bit bus

- even port number: D0-D7
- odd port number: D8-D15
- connect A0, A1 to system A1, A2, and access even address

4.0.6 Output Port Design

- latch data from CPU

4.0.7 Input Port Design

- use tri-state buffer (one-way l, one-way r, bi)

5 Example: 8255

5.1 Packaging

- PA x 8 (Port A), connect to I/O device
- PB x 8 (Port B), (8-bit for a port, so calls parallel interface)
- PC x 8 (Port C)
- Data Bus (8-bit)
- WR, RD, RESET, Vcc, GND, CS
- A0, A1 (00, 01, 10 for A, B, C, and 11 for control)

5.2 Internal Structure

- Group A = Port A + Port C upper
- Group B = Port B + Port C lower
- Port A,B are programmable all I/ all O
- Port C can be split into two separate PCU and PCL
- Port C ports can be programmed individually
- Control Register (CR), set up the chip
- Data bus buffer (bidirectional, tri-state, 8-bit)
- R/W control logic
  - RESET: high-active, clear CR, all ports input
  - refer to slides for table of CS, A0, A1, RD, WR → function
  - not chip selected - float

5.3 Operation Modes

- I/O modes
  - Mode 0, simple I/O
    - PA, PB, PC: PCU(PC4-PC7), PCL (PC0-PC3)
    - no handshaking
    - each port can be programmed as I/O

  - Mode 1
    - PA, PB as I/O with handshaking
    - PCU(PC3-PC7), PCL(PC0-PC2) as handshake lines
  - Mode 2
    - PA for bidirectional handshake data transfer
    - PCU(PC3-PC7) as handshake
- BSR mode
  - Port C
  - Only PC used as output
- set with CR
  - A1=1, A0=1 to select
  - D7=1, I/O mode
    - D6-D3, Group A
      - D6-D5, 00 Mode 0, 01 Mode 1, 1* Mode 2
      - D4=1, PA Input, D4=0, PA Output
      - D3 PCU I/O
    - D2-D0, Group B
      - D2=0,1 Mode
      - D1 PB
      - D0 PCL
  - D7=0, BSR mode
    - `MOV AL, 00000010B; OUT`
    - do the same in I/O mode
      - `IN AL, CPORT`
      - `AND AL, balahbalah`
      - `OUT CPORT, AL`

5.3.1 Mode 0

- no handshaking (e.g. switch)
- input data not latched, output latched

5.3.2 Mode 1

- Strobe I/O
- PA, PB I/O
- PCU for PA handshake

- PCL for PB handshake
- I/O latched
- handshake signals
  - STB: input signal, load data
  - IBF: input latch contains information (for programmed I/O)
  - INTR: request an interrupt (for interrupted I/O)
- input
  - PC6, PC7 can be used as I/O
  - INTE is an interrupt bit
- input timing
  - put data to PA0-PA7
  - activate STBA
  - 8255 activates IBFA
  - IBFA, !STBA, INTEA all high, 8255 activates INTRA
  - !RD clear INTRA signal
  - IBFA signal cleared
- output
  - PC4, PC5 used as I/O
- output timing
  - INTRA active, CPU writes data to PA, which clears INTRA
  - 8255 activates !OBFA
  - output sends !ACKA, makes !OBFA high
  - !OBFA, !ACKA, INTEA high, 8255 sends INTRA

5.3.3 Mode 2

- PC2-0 for I/O
- refer to slides
- what causes INTRA? check !OBFA and IBFA

6 Example: Timer

- 8253/54 programmable interval timer
- generate a lower frequency for various uses
- usage
  - event counter

    - accurate time delays (software delay not prone to interrupt)
- gate used to enable or disable counter
- square wave, one shot
- duty cycle = high / cycle

6.1 Internal Structure

- 16-bit down counter
- 8254: DC to 10MHz (8253: 2.6MHz)
- 8254 support read back
- reciting data and port is really boring, just refer to slides

6.2 Mode 0

- interrupt on terminal count

6.3 5 Modes

- refer to slides