ECE/CE 3720: Embedded System Design

Chris J. Myers
Lecture 2: I/O Ports

Fanout Requirements

\[ \text{fan-out} = \min(\frac{I_{OH}}{I_{IH}}, \frac{I_{OL}}{I_{IL}}) \]

I/O Currents

<table>
<thead>
<tr>
<th>Family</th>
<th>( I_{OH} )</th>
<th>( I_{OL} )</th>
<th>( I_{IH} )</th>
<th>( I_{IL} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard TTL</td>
<td>0.4mA</td>
<td>16mA</td>
<td>40( \mu )A</td>
<td>1.6mA</td>
</tr>
<tr>
<td>Schottky TTL</td>
<td>1mA</td>
<td>20mA</td>
<td>50( \mu )A</td>
<td>2mA</td>
</tr>
<tr>
<td>Low-power Schottky TTL</td>
<td>0.4mA</td>
<td>4mA</td>
<td>20( \mu )A</td>
<td>0.4mA</td>
</tr>
<tr>
<td>High-speed CMOS</td>
<td>4mA</td>
<td>4mA</td>
<td>1( \mu )A</td>
<td>1( \mu )A</td>
</tr>
<tr>
<td>68HC11</td>
<td>0.8mA</td>
<td>1.6mA</td>
<td>1( \mu )A</td>
<td>1( \mu )A</td>
</tr>
</tbody>
</table>

Voltage Thresholds

(See Figure 1.56)
Transistor Implementations

(See Figures 1.58-1.60)

Open-Collector Gates

(See Figures 1.61-1.62)

- $R \leq (+5 - V_{out})/I_{out}$

Memory-Mapped Computer System

(See Figure 1.3)

Isolated I/O Computer Systems

(See Figure 1.8)
Read/Write Cycles

(See Figures 1.4 and 1.5)

DMA Read/Write Cycles

(See Figures 1.6 and 1.7)

Definitions for I/O Ports

PORTA equ $1000 PA7 i/o, PA6-PA3 outputs, PA2-PA0 inputs
PACTL equ $1026 Bit 7 specifies whether PA7 is i/o
PORTB equ $1004 PB7-PB0 are all readable outputs
PORTC equ $1003 PC7-PC0 can be input or output
DDRC equ $1007 Direction register for Port C
...
#define PORTA *(unsigned char volatile *)(0x1000)
#define PACTL *(unsigned char volatile *)(0x1026)
#define PORTB *(unsigned char volatile *)(0x1004)
#define PORTC *(unsigned char volatile *)(0x1003)
#define DDRC *(unsigned char volatile *)(0x1007)
...

Input Ports

(See Figures 1.64 and 1.65)
Slide 13

Readable Output Port

(See Figures 1.66)

Slide 15

Software to Read Port A and Write Port B

NotGate ldaa PORTA Read from Port A into Reg A
com PORTA Logical complement
sta PORTB Write from Reg A to Port B
bra NotGate

void main(void) { unsigned char data;
while(1) {
    data = PORTA; // Read from Port A
    data = ~data; // Logical complement
    PORTB = data; // Write from Reg A to Port B
}}

Slide 14

Bidirectional Ports

(See Figures 1.67-1.69)

Slide 16

Simple I/O Software

clr DDRC Set Port C to input DDRC = 0x00;
ldaa PORTC Read Port C into Reg A H = PORTC;
sta PORTC Store result into memory
ldaa #$0F Load all 1’s into Reg A DDRC=0x0F;
sta DDRC Set Port C to output

Init staa DDRC void Init(unsigned char Value) {
    rts DDRC=value; }

Set staa PORTC void Set(unsigned char Value) {
    rts PORTC=value; }

Read ldaa PORTC void Read(unsigned char Value) {
    rts return(PORTC); }
**I/O Example**

(See Figures 1.71)

```c
void init(void){
    DDRC = 0xF0; // PC7-PC4 outputs, PC3-PC0 inputs
}
void main(void){
    init(); // call ritual once
    while(1){
        data=PORTC; // input
        data=(~data)<<4; // complement and shift
        PORTC=data;}} // output
```

---

**A 4-bit NOT Gate**

```
PORTC equ $1003 ;I/O port
DDRC equ $1007
init ldaa #$F0 ;PC7-PC3 out
       staa DDRC ;PC3-PC0 in
       rts
org $E000 ;ROM
main lds #$00FF ;SP=$00FF
       bsr init ;ritual
loop ldaa PORTC ;input
       coma ;logical not
       lsla ;shift
       lsla
       lsla
       lsla
       staa PORTC ;output
       bra loop ;repeat
org $FFE
fdb main ;reset vector
```