Memory-Mapped I/O

- To a processor, everything is memory.
- Peripherals appear as magical memory locations.
- Status registers: when read, report state of peripheral.
- Control registers: when written, change state of peripheral.

Typical Peripheral: PC Parallel Port

- **D7**
  - Busy
- **D6**
  - Ack
- **D5**
  - Paper
- **D4**
  - Sel
- **D3**
  - Err
- **D2**
  - Sel Init
- **D1**
  - Auto
- **D0**
  - Strobe

1. Write Data
2. Assert Strobe
3. Wait for Busy to clear
4. Wait for Acknowledge

A Parallel Port Driver

```c
#define DATA 0x378
#define STATUS 0x379
#define CONTROL 0x37A
#define NBSY 0x80
#define NACK 0x40
#define OUT 0x20
#define SEL 0x10
#define NERR 0x08
#define STROBE 0x01
#define INVERT (NBSY | NACK | SEL | NERR)
#define MASK (NBSY | NACK | OUT | SEL | NERR)

void write_single_character(char c) {
  while (NOT_READY(STATUS)) ;
  outb(DATA, c);
  outb(CONTROL, control | STROBE); /* Assert STROBE */
  outb(CONTROL, control ); /* Clear STROBE */
}
```

Interrupts and Polling

- **Polling:** “Are we there yet?”
- **Interrupts:** Ringing Telephone

Interrupts

Basic idea:

1. Peripheral asserts a processor’s interrupt input
2. Processor temporarily transfers control to interrupt service routine
3. ISR gathers data from peripheral and acknowledges interrupt
4. ISR returns control to previously-executing program

Many Different Interrupts

Processor receives interrupt
ISR polls all potential interrupt sources

Interrupt Polling

- Processor
  - Peripheral
  - Peripheral
  - Peripheral

What’s a processor to do?

Intel 8259 PIC

Prioritizes incoming requests & notifies processor ISR reads 8-bit interrupt vector number of winner IBM PC/AT: two 8259s; became standard