2020-02-05

Meet the 8051

- Some iterations of the 8051 have included ROM/RAM, some don't
- Upon powering on, the 8051 will fetch the first instruction from the ROM

Special Function Registerstt

Memory bank switching

- Some 8051s have an additional 128 bytes of RAM which can be accessed using a "indirect address" syntax, where the requested address is placed in a spare register and prepend the register name with an @ to access the memory address

---

2020-02-06

Interface to RAM/ROM

- Address bus (16 bit in our case)
- Data bus (8 bit in our case)
- Control signals
  - Chip Enable (CE) must be activated to use any operation
  - Output Enable (OE) must be activated to read data from address onto data bus
  - Write (WR) must be activated to write data on data bus to address
  - Control signals are sometimes "active-low", meaning that you connect them to ground to make them active
- In order to drive address and data bus, we sacrifice two IO ports to communicate with the chip
  - If we weren't using external ROM/RAM, we would be able to use it for GPIO
  - We can actually use one port for both the data bus and half of the address bus by multiplexing the bus and using a latch to save the lower address byte while the data is read
- External Access (EA) is an active low signal that communicates to the 8051 that the program should be found on external memory

ROM vs RAM usage by 8051

- Program Store Enable (PSEN) is an active low signal that gets activated when the 8051 is fetching some instruction data

- Write (WR) and Read (RD) are active low signals that get activated when the 8051 is attempting to write or read to external RAM

2020-02-11

Review of RAM/ROM trick

The ROM contains minmon, starting at address 0 in the ROM. When the switch is set to minmon mode, the ROM is mapped into the lower half of the address space, and the RAM is mapped into the upper half.

Downloading code to minmon using the serial port puts the binary into the upper half, starting at 0x8000 (which is where the RAM is mapped when the switch is in minmon mode). You can then run the code one of two ways:

1. Issue a goto command to jump program execution to 0x8000
    - This will work as long as you've been careful to ensure that any absolute memory addresses are aligned to that address (i.e. don't have jumps that will put you back into the middle of minmon)
2. Reset the board and switch into run mode, so that RAM is mapped into 0x0000 and your program will not have to obey any special discipline about absolute memory addresses

Serial Port

There are an incredible number of communication schemes available for embedded systems, but one of the simplest and oldest is RS232 (serial port).

The protocol is **asynchronous**, which means that no clock is sent with the signal; instead, the client and host must first agree on a **baud rate** that specifies how many bits will be transmitted each second.

The protocol synchronizes the host and client by using a start bit and a stop bit to begin and end a transmission

Hardware on the 8051 to Enable Serial Port Usage

Two buffers exist on the 8051 for storing serial data: the **Serial Buffer Transmitter** and the **Serial Buffer Receiver**. Both are referred to as **SBUF**, but the former can only be written to and the latter can only be read from, so the names actually don't conflict.

There are several control registers that need to be set to configure the serial port:

- **TH1** sets which baud rate setting should be used

- **SCON** configures information about what will be sent/received (e.g. to optionally enable 9-bit transmission)
- **TMOD** configures timers 1 and 0, setting modes about interrupts
- **TCON** configures more information about interrupts

2020-02-25

Interrupts

Interrupts are used to briefly transfer execution to a handler routine when some event happens. Each interrupt corresponds to a location in the **interrupt vector**, which is just a region in memory where interrupt sources are programmed to jump to.

Polling vs Interrupts

Choosing whether to deal with I/O using polling or interrupts is a trade-off involving overhead and latency. For very frequent events, polling is usually wiser, because the overhead of the effective function call that is done to save the return address. For infrequnet events, that overhead is negligible compared to the overhead you would spend polling, especially if you need a low latency.

Timing of interrupts

An interrupt will not be accepted iff

- A higher priority ISR is being executed
- The current instruction being executed is a RETI
- The current instruction isn't on the last machine cycle of that instruction