CS 342302 Operating Systems

Fall Semester 2021

Prof. Pai H. Chou

Weekly Review 1

The questions here serve the purpose of reviewing concepts from the lecture, and expect the concepts to be tested on the midterm and final. However, they are by no means exhaustive. Anything covered in the lecture and projects can be tested.

1. Definitions and Short Answers - week 1 (9/13 lectures)

1. What is **batch processing**? What are its advantages? Disadvantages?

A: Batch processing is the processing or transactions in a group or batch. No user interaction is required once batch processing is underway.

Advantages:

* Repeated jobs are done fast without user interaction.
* Offline processing causes less stress on the processor.
* Sharing systems for multiple users.
* You can assign specific time for the batch jobs.

Disadvantages:

* One job at a time.
* No interaction between users and jobs.
* CPU is often idle. I/O speed much slower than CPU speed.

1. What is **multiprogramming**? What disadvantage of batch processing does it address?

A: Multiprogramming allows several jobs to be loaded into memory. It also allows multiplexed among jobs, which is one of the main disadvantages of batch processing.

1. Compare **multiprogramming** and **multitasking** in terms of number of users, number of jobs running, and need for support features. (Textbook p. 23)

A:

1. What is an **instruction set architecture** (ISA)? How is it different from a CPU or a processor?

A: An instruction set architecture (ISA) is an abstract model of a computer. A CPU or processor is the device that executes the instructions described by the ISA. It is the implementation of the ISA.

1. What are reasons for the trend from single processor to multiprocessor architectures?

A: The reasons include but are not limited to increasing throughput, economy, reliability and power.

1. What makes tightly coupled multiprocessors difficult to **scale** to many processors?

A:

1. What are examples of **real-time** systems? How do they differ from **high-performance** systems?

A:

1. What are examples of **hard real-time** vs **soft real-time** systems?

A:

2. EdSim51 and 8051 - week 1 (9/15 lecture)

1. What is an assembler?

A: Is a type of computer program that interprets software programs written in assembly into machine language, code and instructions that can be executed by a computer.

An assembler enables software and application developers to access, operate and manage a computer’s hardware architecture and components.

1. Given the sample assembly program:  
   ORG 0000H  
   MOV 90H, #24H  
   END
   1. What is a **directive** in this assembly program, and how is it different from an instruction?

A: A directive gives an instruction to an assembler, also known as assembler directives.

* 1. What is an **instruction**? What does the assembler do to an instruction?

A: An instruction tells the processor what to do, they correspond to machine instructions.

* 1. What is an **opcode** in the above example? An **operand**?

A: An opcode specifies the operation or function to execute. In the example above **MOV** is an opcode. An operand on the other hand, are the “arguments” to an opcode. The operands in the above example are 90H and #24H.

1. What does 90H refer to? What does #24H refer to? Why is there a # in front of 24H but not in front of 90H?

A: 90H represents the address of special function register. #24H is represents the hex value 0x24 where the # sign indicates immediate data or the equivalent of a constant value.90H represents a direct address.

1. What is a **NOP**? and how do you pronounce it?

A: It is an operation that does nothing. Execution continues with the next instruction. No registers or flags are affected by this instruction. Generally used to generate a delay in execution or to reserve space in code memory. It is pronounced “NO – OP”

1. Why is 8051 called a **Harvard architecture**? How is it different from a **von Neumann** architecture?

A: In a Harvard architecture, code and data memories have separate address spaces. In a Von Neumann architecture this separation does not exist however it has a 32—bit address space which is bigger.

1. What is a **PC** in a processor? What are the ways PC value can change?

A: A PC is a program counter in a processor. It is the address of the current instruction. A PC can change by jump and branch instructions.

1. How many bytes is 8051’s **code memory**? How many bits are needed to represent the code address?

A: 8051’s code memory is 64 KB. 8 bits are required to represent the code address.

1. What is **DPTR** in 8051? How is it related to **DPL** and **DPH**?

A: Is a 16-bit general purpose register used as a data pointer. DPL and DPH are concatenated to form DPTR. DPTR -> “Data Pointer”

1. How big is the IDATA memory in 8051?

A: The Internal Data Memory in the 8051 is 256 bytes. The first 128 bytes of IDATA are directly and indirectly addressable (0x00 to 0x7H). The upper 28 bytes (Addresses 0x80 to 0xFF) can be addressed only indirectly and correspond to SFR’s. At 20H there

1. What is a special-function register (SFR)?

A: A special function register stores values but causes side effects. For. Example in 8051 if we want to write to port P1 we must

1. What is the meaning of **simplex**, **half-duplex**, and **full-duplex** communication? Which one is the UART (serial port)?

A: Simplex: One direction of communication only (R or T).

Half-duplex: One direction at a time in a two-way communication supported (RT).

Full duplex: Two-way communication supported with two directions (RT) simultaneously.

1. If #24H refers to hex 24 integer value (“immediate”) in Intel assembly, why #FFH does not refer to hex FF integer value? Name two ways to express 0xFF in Intel assembly syntax.

A: Because of the limitations of the time, a pound sign must be followed by a number (which represents a literal value) or a symbol which represents a label, so #FFH means the literal value pointed by label “FFH”.

You can express 0xFF in Intel Syntax as #0FF (Add a useless 0) or simply write it as #-1.

1. What is the meaning of MOV A, 17 in 8051 assembly, and how is it different from MOV A, #17 ? What about MOV A, 17H ? MOV A, #17H ?

A: MOV A,17 means move whatever data is in decimal address 17 (0x11) of IDATA memory into the accumulator A. This is an example of direct addressing mode. On the other hand, MOV A, #17 moves the decimal value 17 (0x11) into accumulator. Nothing the appended H, The other two instructions do exactly the same but with address and hexadecimal values 17 (0x17).

1. What is a **general-purpose input/output** (GPIO) port? What are they called on 8051?

A: The GPIO ports handle both incoming and outgoing digital signals. The 8051 MCU has 4 8-bit I/O ports each with 8 pins. Thus a total of 32 I/O pins allow the MCU to be connected with peripheral devices.

1. Why do you have to write a **0 bit** to **turn on** an LED segment? Why write a **1 bit** to **turn off**?

A: Bit 0 means the led segment is connected to ground (0 = provide ground) while Bit 1 means the led segment is not connected to ground and thus cannot turn on (1 = do not provide ground).

1. What is an “immediate” operand?

A: An immediate operand, for example, #17His a literal value.

1. What is a "direct" operand?

A: A direct operand, for example, 17H is an IDATA address.

1. How do you pronounce “UART”? What is another more descriptive term for UART?

A: It is pronounced “U-ART”. Another more descriptive term for UART is Universal Asynchronous Remote Transmitter.