Registers in Microcomputers

Lecture # 2

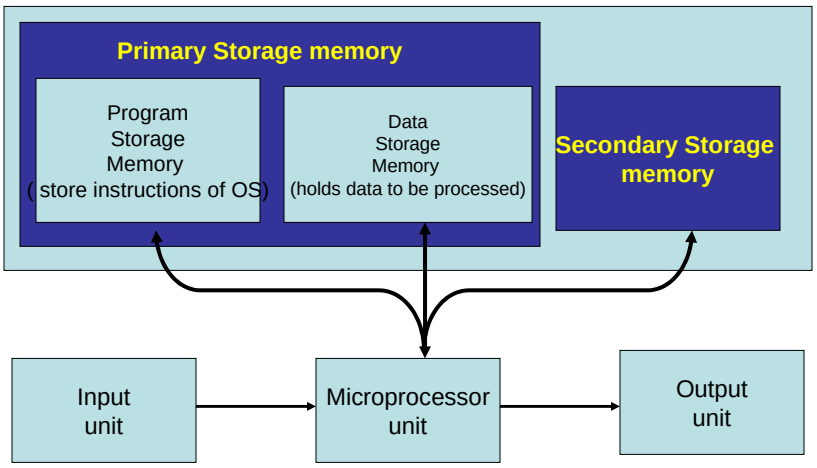
Main, mini & micro

- Mainframe
  - Used in the environment where it serve for large number of users
    - University or where large data processing is a requirement
- Minicomputer
  - Large number of users accessing some central machine through dumb terminal
  - Not self sufficient computers, if central machine is not working no terminal will also be working
- Microcomputer
  - Known as PC, designed for used by individual as standalone or through connected through LAN
  - It is a general purpose electronic data processing machine

General Architecture

- The hardware of a microcomputer system can be divided in four functional sections
  - The input unit
  - The microprocessor unit
  - The memory unit
  - The output unit
- Each unit has a special function
- The communication medium that is linking these units is bus
  - The address bus
  - The data bus
  - The control bus

General Architecture of microcomputer

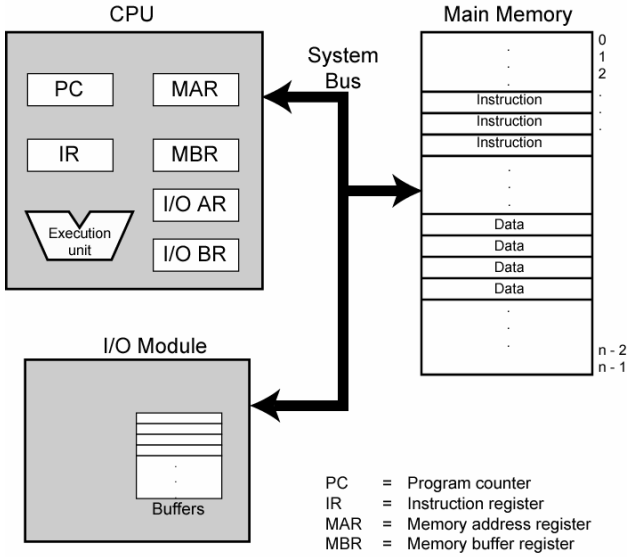


The heart of microcomputer system, implemented with a VLSI device – " Microprocessor or just Processor " A general purpose processing unit built into a single IC

Basic Computer components

- Components are:
  - Registers
  - Buses
  - Memory
  - CPU
  - Peripherals

Computer Components: Top Level View



Recall

- Sequential Logic?
  - Combinational logic:
    - "forgets" its results when the inputs are no longer available
  - Sequential logic:
    - "remembers" results until the next *clock signal*
- Latch vs. Flip Flop
  - Latch changes by level
    - As long as C is high, Q follows D
  - Flip flop changes by edge
    - Q takes value of D at rising (or falling) edge only

Latches and Flip-Flops

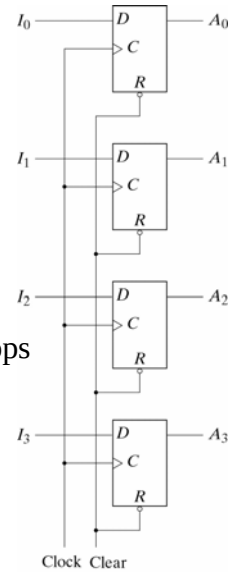
- In the same way that gates are the building blocks of combinational circuits, latches and flip-flops are the building blocks of sequential circuits.
- Gates had to be built directly from transistors, latches can be built from gates, and flip-flops can be built from latches.
- The difference between a latch and a flip-flop is that a latch does not have a clock signal, whereas a flip-flop always does.

Registers

- Registers → a common sequential device:
  - They are frequently used in building larger sequential circuits.
- A register is a collection of flip-flops
  - basic function is to hold information
- Registers hold larger quantities of data than individual flip-flops.
  - Registers are central to the design of modern processors.
- One register is simply a set of D flip-flops, one per bit
  - Data inputs are D's
  - Data outputs are Q's and Q's
  - Clocks all tied together

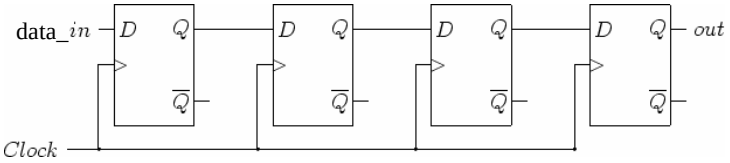
4-Bit Parallel Registers

- The common clock input triggers all flip-flops on the positive edge of each pulse → the binary data available at the 4 inputs are transferred into the register.
- The four outputs can be sampled to obtain the binary information stored in the register.
- When the clear input *R* goes to zero, all flip-flops are reset (register is cleared to 0's).
- 8 –Bit Registers can store a byte



Shift Registers

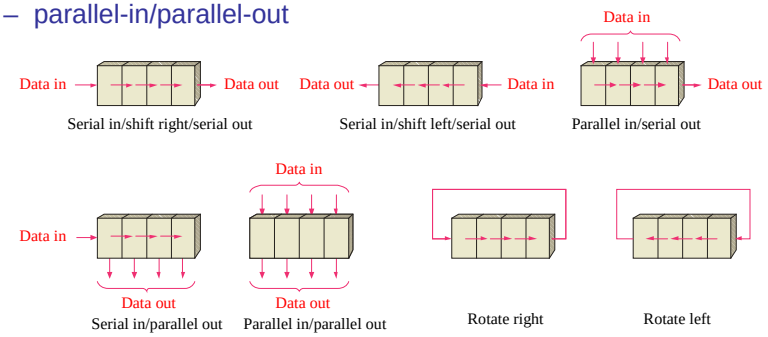
A *Shift Register* is capable of shifting its binary information in one or both directions.



On the leading edge of the first clock pulse, the signal on the data_in is latched in the first flip-flop. On the leading edge of the next clock pulse, the contents of the first flip-flop is stored in the second flip-flop, and the signal which is present at the data_in is stored is the first flip-flop, *etc.*

Shift Registers

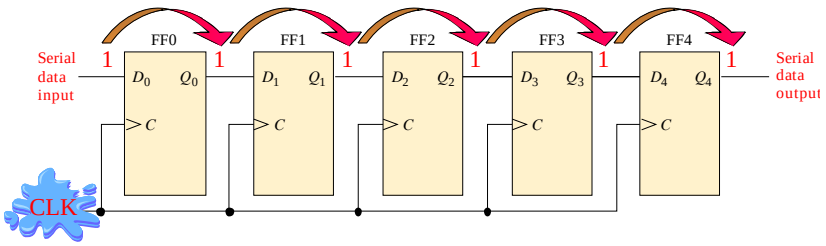
- A shift register is an arrangement of flip-flops with important applications in storage and movement of data.
- A shift register is a register that moves information on the clock signal
  - serial-in/serial-out
  - serial-in/parallel-out
  - parallel-in/serial-out
  - parallel-in/parallel-out



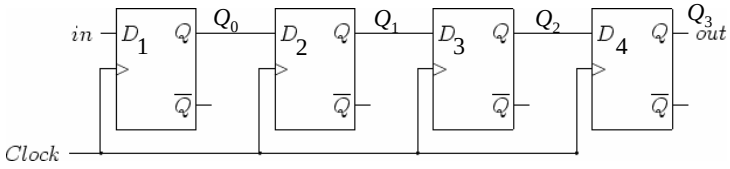
Serial-in/Serial out Shift Register

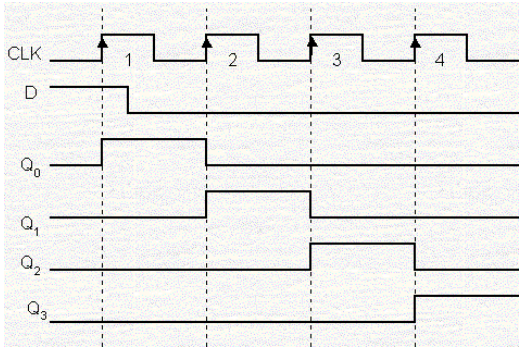
Shift registers are available in IC form or can be constructed from discrete flip-flops as is shown here with a five-bit serial-in serial-out register.

Each clock pulse will move an input bit to the next flip-flop. For example, a 1 is shown as it moves across.

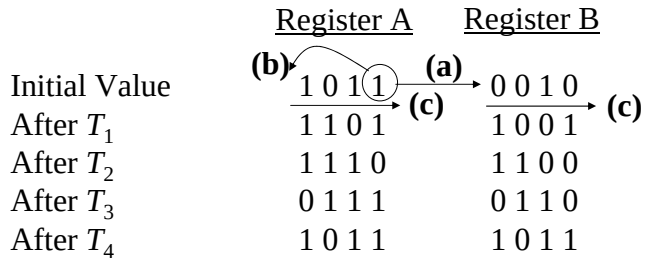


Serial Shift Registers – Timing Diagram





Serial Transfer Example



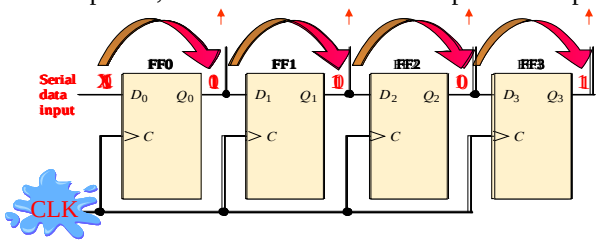
With the first pulse T_1, (a) the rightmost bit of A is shifted into the leftmost bit of B and (b) also circulated into the leftmost position of A. At the same time, (c) all bits of A and B are shifted one position to the right.

A Basic Application

An application of shift registers is conversion of serial data to parallel form.

For example, assume the binary number 1011 is loaded sequentially, one bit at each clock pulse.

After 4 clock pulses, the data is available at the parallel output.

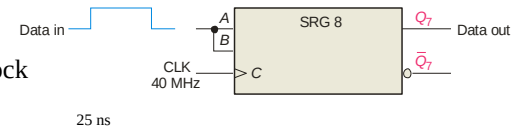


Shift Register Applications

Shift registers can be used to delay a digital signal by a predetermined amount.

Example An 8-bit serial in/serial out shift register has a 40 MHz clock. What is the total delay through the register?

Solution



The delay for each clock is 1/40 MHz = 25 ns

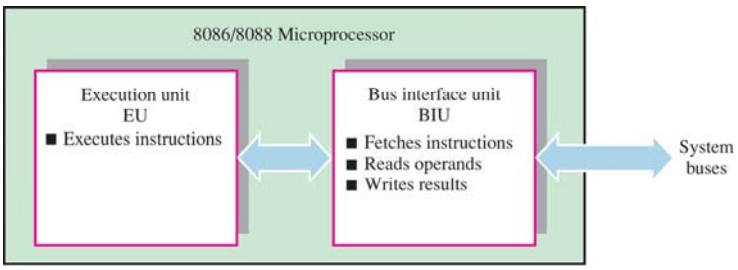
The total delay is 8 x 25 ns = 200 ns

8086 Architecture

- The microprocessors functions as the CPU in the stored program model of the digital computer.
- The microprocessor also has a S/W function.
  - It must recognize, decode, and execute program instructions fetched from the memory unit.
    - Fetch – Decode – Execute – Store Cycle
  - This requires an Arithmetic-Logic Unit (ALU) within the CPU to perform arithmetic and logical (AND, OR, NOT, compare, etc) functions.
- The 8088/ 8086 CPUs are organized as two processing units, called the
  - Bus Interface Unit (BIU) and the Execution Unit (EU).
    - Each unit has a dedicated function and both operate at the same time

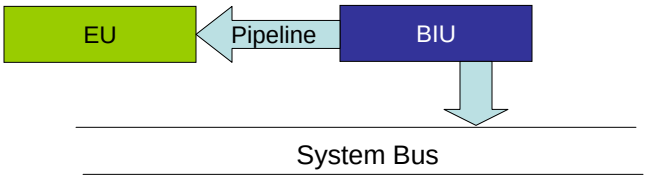
Microprocessor Architecture

- The 8086/8088 has two separate internal units, the EU and the BIU
  - The microprocessors functions as the CPU in the stored program model of the digital computer.
    - Its job is to generate all system timing signals and synchronize the transfer of data between memory, I/O, and itself.
    - It accomplishes this task via the three-bus system architecture.



8086 Architecture

- The BIU provides H/W functions, including
  - generation of the memory and I/O addresses for the transfer of data between the outside world -outside the CPU and the EU.
- The EU receives
  - program instruction codes and data from the BIU,
  - executes these instructions, and
  - store the results in the general registers.

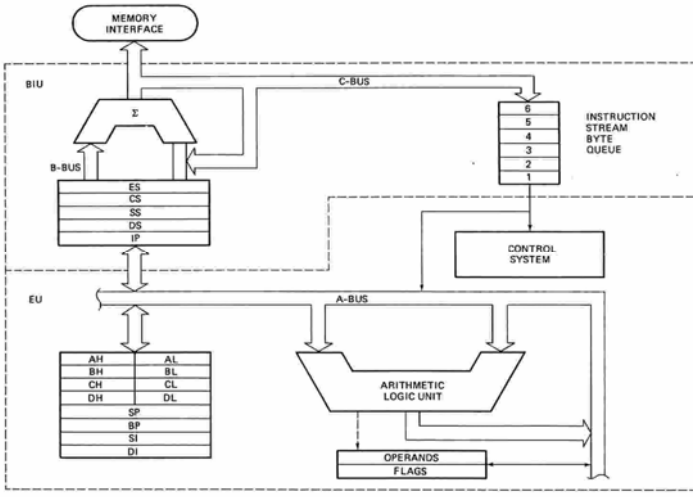


8086 Microprocessor

- 8086 is a 16 bit microprocessor
  - ALU
  - Internal registers and
  - Most of the instructions are of 16 bit binary word
- It has a 16 bit data bus- read or write 16 or 8 bit at a time to memory or to port (8bit bus for 8088)
- It has 20 bit address bus- can address 1 Mbyte of memory locations

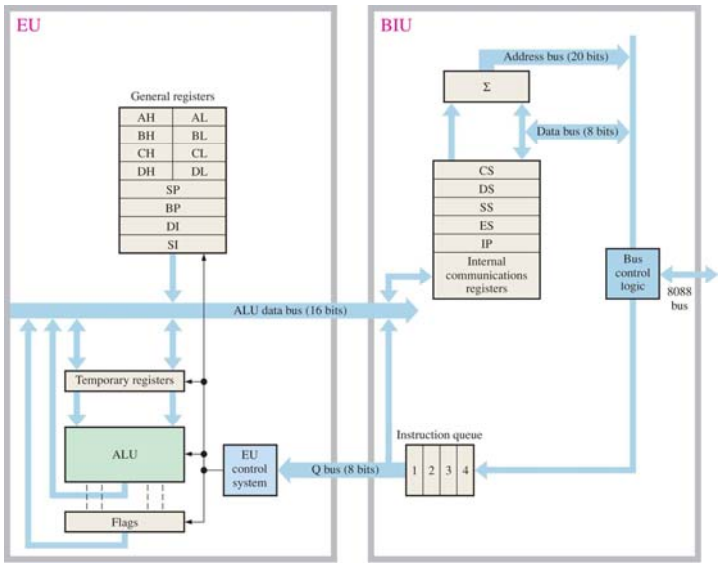
Architecture

A typical x86 Architecture



Intel 8086 Architecture, the 1st member of x86 family

Microprocessor Architecture



8086 Registers

- The 8086 microprocessor has a total of **fourteen** registers that are accessible to the programmer.
  - Eight of the registers are known as **general purpose** registers located in EU, these can be used by the programmer for data manipulation.

Gen Purpose Registers

- EU contains eight Gen purpose registers as AH, AL, BH, CH, CL, DH and Dl
  - The first four registers are sometimes referred to as **data** registers.
    - Used individually for temporary storage of 8-bit data
    - Can be used to store 16-bit data, if grouped together as
  - AL and AH ref to as AX (Accumulator) used to hold temp results after arithmetic operation
  - BL and BH ref to as BX (Base) used to hold base address of data located in the memory
  - CL and CH ref to as CX (Count) to count certain loop instr eg shift counts
  - DL and DH ref to as DX (Data) gen purpose register holds data

8086 Registers

- The four **index** registers each of the registers is **16 bits long** i.e. can contain a 16-bit binary number can be used for
  - arithmetic operations but their use is usually concerned with the memory addressing modes of the 8086.
    - They are the **sp**, **bp**, **si** and **di** registers.

Pointers and index Registers

- Also general purpose registers, more often used for:
  - arithmetic operations but their use is usually concerned with the memory addressing modes of the 8086.
  - To index or point to memory
    - SP (Stack Pointer) used to address data in LIFO stack memory, when PUSH and POP instructions are executed
    - BP (Base Pointer) 16 bit-used to address an array of data in the stack memory
    - SI (Source index) 16 bit
    - DI (destination index) 16- bit, to hold 16 bit offset data

Instruction Pointer Register

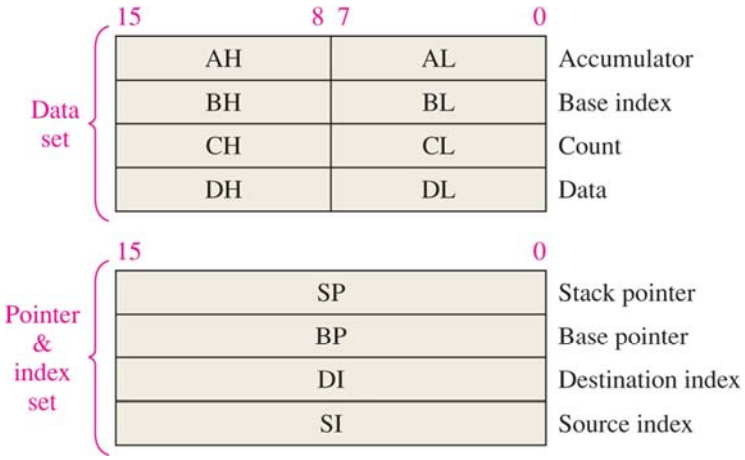
- IP (Instr Pointer) always used to address the next instruction by adding the contents of IP to code segment CS register
- This is a crucially important register
  - used to control which instruction the CPU executes.
  - The **ip**, or *program counter*, is used to store the memory location of the next instruction to be executed.
- The CPU checks the program counter to ascertain which instruction to carry out next.
  - It then updates the program counter to point to the next instruction.
  - Thus the program counter will always point to the next instruction to be executed.

8086 Registers

Segment Registers		
CS	Code Segment	16-bit number that points to the active code-segment
DS	Data Segment	16-bit number that points to the active data-segment
SS	Stack Segment	16-bit number that points to the active stack-segment
ES	Extra Segment	16-bit number that points to the active extra-segment
Pointer Registers		
IP	Instruction Pointer	16-bit number that points to the offset of the next instruction
SP	Stack Pointer	16-bit number that points to the offset that the stack is using
BP	Base Pointer	used to pass data to and from the stack
General-Purpose Registers		
AX 16 bit (AH +AL 8 bit)	Accumulator Register	mostly used for calculations and for input/output
BX (BH + BL)	Base Register	Only register that can be used as an index
CX (BH + BL)	Count Register	register used for the loop instruction
DX (BH + BL)	Data Register	input/output and used by multiply and divide
Index Registers		
SI	Source Index	used by string operations as source
DI	Destination Index	used by string operations as destination

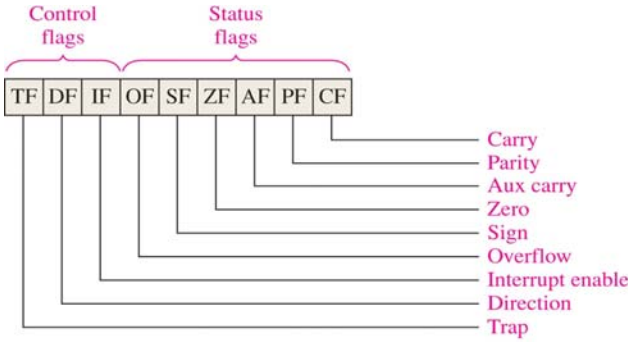
Registers in Microprocessor Architecture

- The general register set



Flag Registers

- Flag is a Flip-flop
  - which indicate some condition produced by the execution of an instruction
    - or Which controls certain operations of Execution Unit (EU)
- 16 bit flag register in EU contains 9 active flags
- The status and control flags



Flag Register

- Six flag are used to indicted some condition produced by an instruction
- These are set and reset by EU on the bases of the result of some arithmetic and logic operation
- These are:-
  - Carry Flag (CF)
  - Parity Flag (PF)
  - Auxiliary Carry Flag (AF)
  - Zero Flag (ZF)
  - Sign Flag (SF)
  - Overflow Flag (OF)

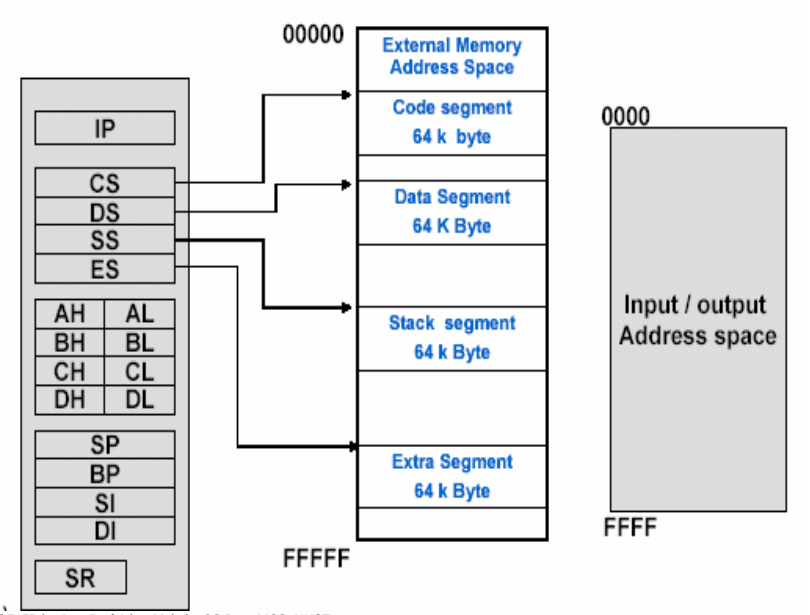
Flag Register

- Remaining three flags are used to control certain operations of the processor
- These are deliberately set and reset flags with specific instructions through program
  - Trap Flag (TF), used for single stepping through program
  - Interrupt Flag (IF), used to allow or block interrupts
  - Direction Flag (DF), used with string instructions
- The **instruction pointer (ip)** and the **status word**, or **flags** register.
  - Neither of these is referenced **directly** by your program.

Segment Registers

- These are unique to x86 microprocessor family, and are designed because
  - All Index and pointer registers are of 16 bit wide, and the memory is of 20-bit (1 M)
  - Requires 20 bit address
- Index and pointer registers are not wide enough to address directly to any memory location
  - To resolve memory is segmented
- Segment of memory
  - A block of 64 Kbytes of memory addressed by a special register called a “segment register”

Software Model of 8086



Segment Registers

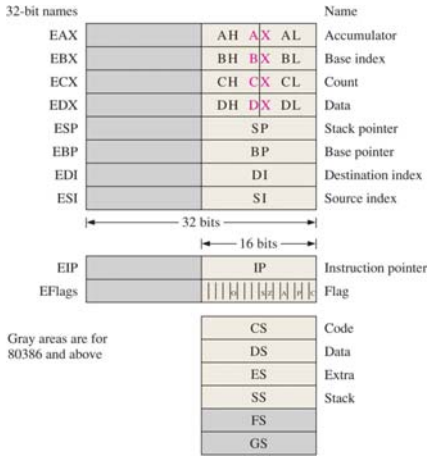
- Within the 1 MB of memory space the 8086/88 defines four 64K-byte memory blocks called the code segment, stack segment, data segment, and extra segment.
  - Each of these blocks of memory is used differently by the processor.
    - The code segment holds the program instruction codes.
    - The data segment stores data for the program.
    - The extra segment is an extra data segment (often used for shared data).
    - The stack segment is used to store interrupt and subroutine return addresses.

Segment Registers - CS, DS, SS and ES

- Store the memory addresses of instructions and data
- Memory Organization
  - Each byte in memory has a 20 bit address starting with 0 to 2^{20}-1 or 1 MB of addressable memory
  - Addresses are expressed as 5 hex digits from 00000 - FFFFF
  - Problem: 20 bit addresses are TOO BIG to fit in 16 bit registers!
  - Solution: Memory Segment
    - Block of 64K (65,536) consecutive memory bytes
      - A segment number is a 16 bit number
      - Segment numbers range from 0000 to FFFF
      - Within a segment, a particular memory location is specified with an offset An offset also ranges from 0000 to FFFF

Microprocessor Architecture

- Registers for the Intel processors from 8086/8088 through Pentium



Instruction fetch and execute

- At the beginning of each instruction cycle
  - Processor fetches an instruction from the memory
    - A register “PC”- program counter holds the address of the instruction to be fetched next
    - Processor always increment the PC after each instruction unless told otherwise
  - The fetched instruction is loaded into a register located in the processor – “ the instruction Register (IR)”
  - The instruction contains bits – to specify the action the processor is to take