**ECC301 Assignment 2**

**Name**: Omar Darweesh

**Student Number**: 20173517

**Department**: Computer Engineering

**Internal organization of 8086 family of microprocessors**

The Intel 8086 microprocessor was initiated and introduced in 1978 which was 3 years later after introducing the Intel 8085, although Intel 8086 was the world’s first **16-bit** processing unit and it was a huge improvement over the older microprocessors which were from the series 8080/8085.

**How did it evolve: -**

The comparison between the older generation to the Intel 8086 is sort of huge since it was the first 16-bit processing unit it had higher capabilities which allowed it to have **1MB** of memory also a bigger address bus than the previous versions which was contained of **20-bit**, it also had a two times as big data bus which consisted of **16-bit** it was also the world’s first pipelined processing unit which was able to of some sorts to multitask while the address bus and data bus are busy sending and receiving data while the central processing unit is processing information, While on the other hand the older Intel 8080/8085 had a way smaller memory which consisted of 64 **KB** and also had a smaller address bus which contained of **16-bit,** it also had a smaller data bus which was only **8-bit** and unfortunately it wasn’t a single chip therefore it wasn’t pipelined.

**Pipelining: -**

In the Intel 8085 microprocessor, the central processing unit could also fetch or execute at a certain time the central processing unit had to get information from the memory then executes it then it would keep doing the same thing to process more information and execute it. But in the Intel 8086 microprocessor which was pipelined it could do both fetching data and executing it simultaneously at the same time while noting that these procedures could be different as well.

Intel added the idea of pipelining but cutting the internal structure of the intel 8088/8086 into two sections which were: -

The execution unit which executes the information given from the memory (EU)

And the bus interface unit which was responsible for sending and receiving data (BIU)

These two sections worked together as one simultaneously the bus interface unit enters the memory while the execution unit executes the information and data which was fetched from the memory this only works if the bus interface unit keeps working ahead of the execution interface because you will need data so you can execute it without the information and data from the random access memory you wouldn’t have anything to execute therefore it has a buffer of queue to keep that in check so the processing and fetching stays as it should be without any mistakes .

If the execution of any information or data takes too long the bus interface unit is going to be filled to the maximum capacity therefore it will render the busses idle it will start sending and receiving data from the memory (fetching) when there is at least 2 byte room in the queue .

When a jump instruction is used the microprocessor has to clear all the queue when a jump instruction is used and or executed the bus interface unit starts fetching information from the recent location in the memory. In this circumstance the execution unit has to wait for the bus interface unit to finish fetching a new instruction from the memory so it can start executing it and this is known as branch penalty. The intel 8086 started the life of the x86 architecture, which soon became Intel's most outstanding line of processors. On the 5th of june, 2018, Intel released a limited-edition CPU celebrating the 40th anniversary of the Intel 8086, called the Intel Core i7-8086K.

The intel 8086 had a resourceful set of instructions which allows you do to functions like multiplication and division with all ease it is also carries two states of behavior, which are minimal and maximal states. Maximal state allows the system to have more than one processing unit and the Minimal state is allows the device to have a singular processing unit.

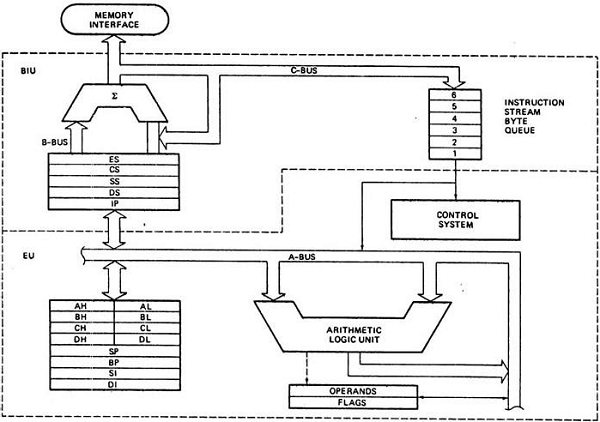
**Intel 8086 characteristics: -**

* The most well-known specifications of the intel 8086 are:
* It got a good instruction queue which is able to keep 6 bytes of instructions in the RAM which is also helpful for faster processing speeds the processor in it self Intel 8086 had three version which were based on the core speed of the processor (frequency).
* As we mentioned before it is a pipelined processor
* During the sending and receiving data or instructions process (fetching) It can fetch as much as 6 bytes of data and keeps the data in the queue
* Using the execution unit that we mentioned in the Pipeline section above it uses the EU to execute those instructions
* It’s a home for 256 aligned interrupts
* It has around 29000 electrical transistors

The table below shows a small comparison between Intel 8085 and Intel 8086

|  |  |  |
| --- | --- | --- |
| **Features** | **Intel 8085** | **Intel 8086** |
| **Size** | 8-bit CPU | 16-bit CPU |
| **Address Bus** | 16-bit Address Bus | 20-bit Address Bus |
| **Memory** | Can have maximum of 64KB Memory | Can have maximum of 1MB memory |
| **Instructions** | No instruction queue | It has an instruction queue |
| **Pipelining** | Not pipelined | Pipelined |
| **I/O** | It can use 2^8 = 256 I/O | It can use 2^16 = 65,536 I/O |
| **Price** | LOW | HIGH |

**The internal architecture of Intel 8086 as a diagram**



We talked about the EU and the BIU before in the previous section which was called pipelining so we are going to skips past them and head over to the ALU

**ALU**: is the unit that works with all arithmetic and also logical functions such as: NOT, OR, AND, -, +, \*, / and so on

**The Flag Register:** it is a unique duty register that and it is also a 16-bit that alternates its status according to the result of the register that all arithmetic and logic results are stored in which is also called the accumulator. The intel 8086 also has 9 flags which are split into 2 categories which are: -

**Conditional Flags**

They show the outcome of the latest arithmetic or logical instruction that has been executed. This is the list of the flags: -

* **The Carry flag**: The following flag alerts and shows a flood circumstance of arithmetic actions and operations.
* **The Auxiliary flag**: utilized as CF however when working with BCD. So, AF will be set when we have flood or sub-current on in BCD estimations. For instance: considering 8-bit ALU unit, Auxiliary flag is on when there is convey from third piece to fourth piece for example convey from lower nibble to higher nibble
* **The Zero flag**: The zero flag is put to one at the time of the consequence of math or legitimate activity is none otherwise it is put as 0.
* **The Parity flag**: is set just when the low-request 8 bits (paying little heed to the operand size) have a significantly number of "1" bits (even equality); it's reset when they have odd equality.
* **The Overflow flag**: the overflow flag speaks to the outcome at the time of framework limit is surpassed.
* **The Sign flag**: The sign flag keeps the indication that the outcome got, for example at the time of the consequence of the activity is negative, at that point the flag is put as 1 otherwise it will be put as 0.

**The Control Flags**

They keep the activities of the EU in check. this is the rundown of all flags that fall under this category: -

* **Trap flag**: which is utilized for singular step control and permits its client to execute each guidance in turn for troubleshooting. On the off chance that it is set, at that point the software could be opened in a solitary advance state which is also called Single Step Mode.
* **Interrupt flag**: which is a hinder empower/incapacitate flag, for instance it is utilized to permit/deny the interference of a software. This is put as 1 for hinder empowered state also put as 0 when the state of interruption is set to disabled.
* **Direction flag**: which is utilized in string activity. At that time, it is set at that point string bytes which are gotten to from the upper RAM addresses to the minimal RAM addresses, it also goes the other way around too.

**General Purpose registers**

We have eight general purpose registers in the intel 8086 which are: AH, AL, BH, BL, CH, CL, DH, and DL. These registers can be utilized exclusively to store 8-bit information and can be utilized two by two to store 16-bit information. The substantial register sets (AL /AH, BL /BH, CL /CH, also DL/DH. also alluded to the CX, BX, AX, also DX individually.

* **CX register**: also alluded to as counting register. It is utilized in circle guidance to save the circle counter.
* **BX register**: which is utilized as a foundation register. Also utilized to keep the beginning foundation location that is RAM region inside the information fragment.
* **AX register**: also, otherwise called as the register in which intermediate arithmetic and logic results are stored which is also named as the accumulator register. It is utilized to keep operands for number juggling activities.
* **DX register**: the DX register is utilized to keep I/O port location for I/O guidance

**The Bus Interface Unit (BIU)**

* The bus interface unit Handles all the exchange of information on the buses for the EU
* The BIU is also responsible for pushing out the addresses
* It also gets directions from the memory so it can carry on with its own functionality
* It is also responsible for perusing information from ports and memory
* And also responsible for composing information to ports and memory

The Bus interface units has 3 parts that work together to keep the BIU in check and out of errors so it keeps the functionality of it seamless and clean and those parts are: -

* Instruction queue
* Segment register
* Instruction pointer

**The instruction queue**

The last segment of BIU is the FIFO gathering of registers called a queue. It is fundamentally a gathering of registers. This plan makes it workable for the BIU to bring the guidance byte while EU is translating a guidance or executing a guidance which doesn't need utilization of buses.

**The Segment Register**

Bus interface unit got four segment buses, for example DS, SS, CS and Extra segment. It keeps the addresses of guidelines & information in the random-access memory, that are utilized by the processing unit so it can reach the RAM areas. Also, additionally it has one pointer register IP, that keeps the place of the following guidance to be executed using Execution Unit.

**ES:** It represents Extra Segment. ES is extra information segment, that is utilized by the string to store and keep the additional location information. (Focuses to the additional fragment wherein information is put away. (in the event that overabundance of 64KB highlighted by the DS). String directions utilize the ES and Destination Index (DI) to decide the 20-piece physical location for the destination)

**CS:** which represents the Code Segment Register.Which is utilized to mark to a RAM area along the code section of what is called Random-Access Memory, where the preformable software is put away**.**

**SS:** which represents the Stack Segment Register. It keeps the memory in check to store information and addresses during execution.

**DS:** which is representing the Data Segment Register. It comprises of information utilized by the program and is gotten to in the information section by a counterbalance mark or the substance of other register which keeps the balance mark. Also, the 16-bit substance of the Source Index (SI) or Destination Index (DI) or then again, a 16-bit relocation are utilized as a balance for figuring the 20-piece physical location

**Note:** all of the registers that we discussed in the previous section are 16-bit registers

**The Instruction Pointer**

Which is also a 16-bit register, that recognizes the area which the following expression of guidance code that will be gotten in the current code segment. Instruction pointer contains a balance rather than the genuine location of the following guidance. The 20-bit address created after expansion of the balance put away in IP to section base location in the code segment is known as the Physical location of the code byte.

**The PIN diagram of the Intel 8086**

The 8086 chip is a 40 pin IC where there are 20 pins on each side of the IC.

**AD0-AD15: Address/Data transport. They are small request address transport. These are flooded with information by a multiplexor. At the point AD lines are utilized to send RAM address the image ‘’A’’ is utilized rather than ‘’AD’’, in an instance A0-A15. At the point when information is sent over ‘’AD’’ lines this image ‘’D’’ is utilized instead of ‘’AD’’, in an instance ‘’D0-D7’’,‘’D0-D15’’, or ’’ D8-D15’’.**

**A16/A19:** top request address transport. During the main clock cycle, it conveys 4-bit address and then it conveys condition waves.

**S2-S1-S0: These are condition pins.** **The following pins are dynamic when T4, T1 and T2 conditions and is come back into an uninvolved condition (1,1,1 when T3 or Tw. They are utilized by the 8288-transport controller to produce all the RAM and I/O activity) get to control waves or alerts.**

**BHE - S7: Transport High Enable/Status. During T1 it is low. It is utilized to empower information onto the biggest ½ portion of data bus, D8-D15. 8-bit gadget associated with upper portion of the information transport use BHE (Active Low) signal. S7 wave is accessible by the time of T2, T3 and T4.**

**RD:** which is utilized for reading activity. This is a yield signal. Which is also enabled on the low circumstance.

**READY:** **which particularly affirmation sent by the RAM or low speed device that they have finished the information move. The sign made accessible by the devices is synchronized by the 8284A clock generator to give prepared contribution to the microchip. The sign is dynamic high**

**INTR: which is also referred to as the Interrupt Request,** **which is an activated analog. It is tested during the final clock patterns of every guidance for deciding the accessibility which the solicitation holds. On the off chance that any intrude on demand is discovered awaiting, the processing unit goes into hinder recognize cycle. Which could be centrally concealed in the wake of coming about the hinder allowance flag. This wave is currently running (set to 1) and has been happening inside.**

**NMI: which is a singular non maskable interrupt, which is a priority activated information that brings about a kind II interfere. A different routine is afterwards vectored inside an interference with the vector query table which is situated in the framework RAM. NMI is not maskable inside by programming. the progress produced using small (zero) to big (one) starts the hinder toward the finish of the present guidance. This information has been copied inside.**

**INTA:** Interrupt recognize. It is dynamic low (0) during T2, T3 and Tw of each hinder recognize cycle.

**MN/MX:** Minimum/Maximum. The following pin wave demonstrates which state the processing unit is going to work in.

**LOCK:** It is a functioning low wave. It shows that more framework bus experts were not permitted to deal with the framework transport while LOCK' is functioning low (0). The ‘’LOCK’’ sign is staying dynamic till the next instruction or expression is done and finished.

**RQ/GT1, RQ/GT0:** Request/Grant. The following pins are utilized by neighborhood transport experts utilized to drive the chip to discharge the nearby transport toward the finish of the microchip's present transport cycle. Every one of the pins has got 2 directions. GT0/RQ got bigger need than GT1/RQ

**TEST:**which is analyzed using a 'Pause' expression. During the event of the TEST pin goes small (0), preforming of instructions will proceed, otherwise the processing unit stays as inactive phase.

**RESET:** This pin requires the microchip to end its current action right away. The sign must be dynamic high (1) for at any rate four clock cycles.

**CLK:** which is also called Clock Input. Which also gives that essential planning to preparing activity and transport control movement.

**QS1, QS0:** Queue Status. These signs demonstrate the status of the inside 8086 guidance line as indicated by the table demonstrated as follows

|  |  |  |
| --- | --- | --- |
| QS0 | QS1 | State |
| ‘0’ | ‘0’ | Not functioning |
| ‘0’ | ‘1’ | Clear the queue |
| ‘1’ | ‘0’ | 1st byte of the operation code from queue |
| ‘1’ | ‘1’ | incoming information from line |

**DT/R:** Data Send/Receive. The following pin needed inside small devices, which need to utilize a 8286 / 8287 information transport handset. That heading of information stream is kept in check using the handset.

**DEN: which is also referred to as Data Enable. is utilized for information empower. This is a functioning low pin that implies at whatever point a 0 is available at this pin then the handset gets empowered and it isolates the information from the multiplexed address and information transport.**

**HOLD/HOLDA:** if an outside device allows this pin then the processing unit quits entering the buses instantly after the ongoing assignment is done.

**ALE:** which is also called as the Address Latch Enable. Address Latch Enable is a shortening for address hook empower. At whatever point a address is available in the multiplexed address and information transport, at that point the chip empowers this pin.

Since we talked about the Structure and the interior design of the Intel 8086 microprocessor, I would like to share a piece of history about that certain microprocessor which was a huge step in the evolution of our current processing units which helped the developers understand microprocessors more and more even though it had a huge power consumption during the 80s because they didn’t have strong power supplies that can help them achieve higher clock speeds and more performance so they had to introduce and initiate the Intel 8088 which was a slight upgrade from the 8086 the main reason behind it was to lower the power consumption of the microprocessor chip so they can also lower the price of the chip. Is the 8086 is used till this day? you may ask and I would say. Nowadays? They aren't utilized anyplace any longer. For some time, the 80186, which was an implanted form of the 8086, had utilizes in many inserted applications as a sensibly savvy arrangement however that dropped out of utilization when the 80386ex was presented.

On the off chance that you signified "Where were 8086 microchips utilized", that would yield an alternate answer. A ton of IBM compatibles utilized the less expensive 8088 variation which had 8 bits outside information transport however there were a couple of PC clones which used the 8086. I feel that the best clone which utilized the 8086 was the Amstrad PC1512/1640/2086 machines and their initial "compact" PC, the PPC640 (which included space inside it for something like 18 D cells to control it).

What's generally astonishing about the gigantic accomplishment of the 8086, however, is the manner by which little individuals expected of it when it was first considered. The historical backdrop of this progressive processor is an exemplary story of how much a little group of brilliant architects can achieve when they're given the opportunity to carry out their responsibilities in creative manners.

At the point when advancement of the 8086 started in May 1976, Intel administrators never envisioned its astounding effect. They considered it to be a minor band-aid venture. They were placing the organization's faith on a profoundly unique and increasingly refined processor called the 8800 (later discharged as the iAPX 432). In a time when most chips despite everything utilized 8-piece information ways, the 8800 would jump as far as possible up to 32 bits. Its progressed performing various tasks abilities and memory-the executive’s hardware would be incorporated right with the CPU, permitting working frameworks to run with significantly less program code.

Be that as it may, the 8800 venture was in a difficult situation. It had experienced various deferrals as Intel engineers found that the unpredictable structure was hard to execute with then-current chip innovation. Furthermore, Intel's issues didn't stop there—it was being outmaneuvered by Zilog, an organization began by previous Intel engineers. Zilog had immediately caught the mid-run microchip advertise with its Z80 CPU. Discharged in July 1976, it was an upgraded clone of Intel's fruitful 8080—the processor that had successfully propelled the PC transformation. Intel still couldn't seem to think of a response to the Z80

ntel executives kept up their confidence in the 8800, yet realized they expected to react to Zilog's danger by one way or another. They went to Stephen Morse, a 36-year-old electrical specialist who had intrigued them with a basic assessment of the 8800 processor's plan defects. The organization's upper metal picked Morse as the sole creator for the 8086. "In the event that [Intel] the board had any notion that this design would live on through numerous ages and into the present ... processors," reviews Morse, "they could never have confided in this undertaking to a solitary individual." (For additional, see our top to bottom meeting with Morse.)

Picking Morse was astonishing for another explanation: He was a product engineer. Beforehand, CPU structure at Intel had been the area of equipment builds alone. "Just because, we were going to see processor highlights from a product point of view," says Morse. "The inquiry was not 'What highlights do we have space for?' however 'What highlights do we need so as to make the product progressively proficient?'" That product driven methodology demonstrated progressive in the business.

Despite the fact that the 8086 was Morse's pet undertaking, he didn't work alone. Joining Morse's group were other Intel workers, including Bill Pohlman, Jim McKevitt, and Bruce Ravenel, every one of whom were basic in putting up the 8086 for sale to the public in the mid-year of 1978.

Past setting out some essential necessities—that the 8086 be perfect with programming composed for the famous 8080 chip and that it has the option to address 128KB of memory—Intel authority avoided Morse's direction. "Since no one anticipated that the structure should live long, no hindrances were set in my direction, and I was allowed to do what I needed," he says.