**United States International University Africa**  
**Nairobi, Kenya**  
**APT2022-Introduction to Assembly Programming**

**PRACTICAL LAB I: 8086 Microprocessor Kit and 8086 Pin Diagram**

**DONE BY GROUP 2 MEMBERS**

**INSTRUCTIONS AND ACTIVITIES**

1. Study the given 8086 microprocessor kit picture of Fig.1.1 and other associated components.

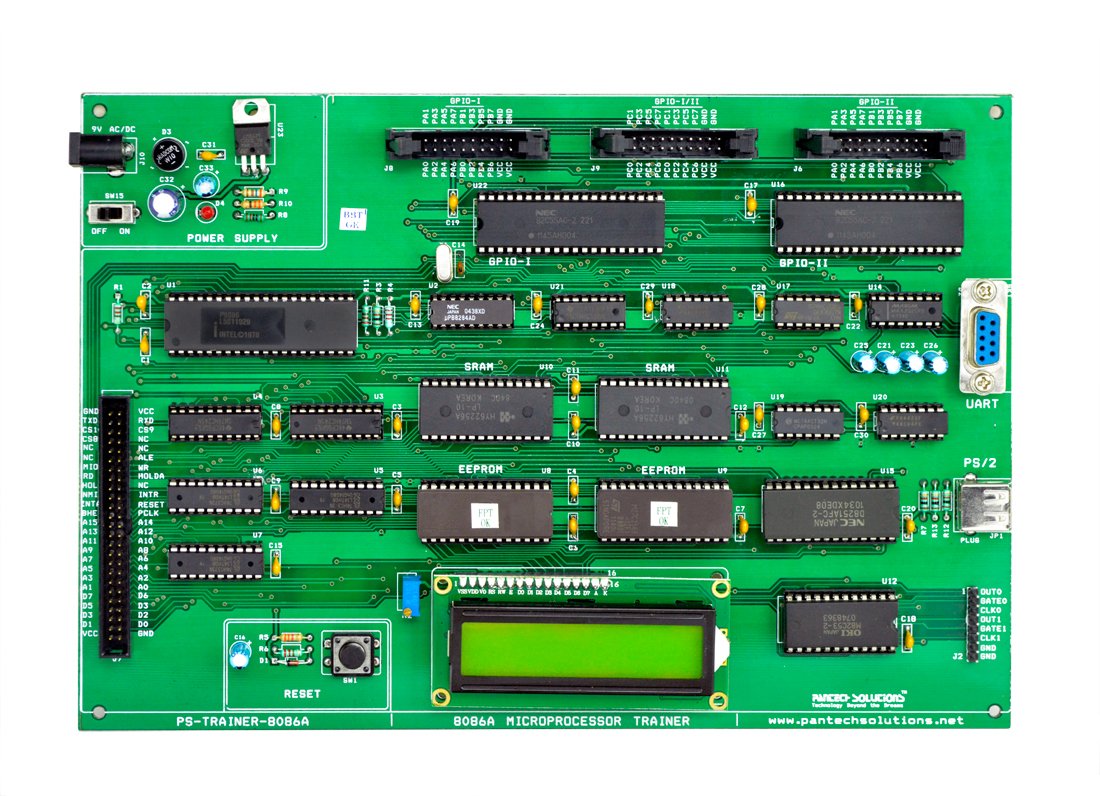


Fig.1 8086 kit picture

1. Identify the components, one by one; in terms of their location in the kit and what purpose they serve. In the photo, label the parts identified, serially e.g., part 1, Part 2, etc.

A piece of paper with writing on it

Description automatically generated with medium confidence

1. Briefly outline the functions of the identified parts in the report write up. For example
2. **p8086A-2-Central Processing Unit (CPU)** - performs arithmetic and logical operations and serves as the “brain” of the computer.
3. **Random Access Memory (RAM) – stores data that is currently being worked on temporarily**
4. **8255 Programmable Peripheral Interfacing devices –** connect peripheral devices to the 8086 microprocessor.
5. **Reset – terminates any current activity that the processor is running**
6. **RS 232** – transmits data between the computer and the microcontroller thereby facilitating communication between them
7. **EEPROM** – is a type of non-volatile ROM that enables individual bytes of data to be erased and reprogrammed.
8. **Time port** - is a clock that governs sequence of events or processes consisting of two timers: timer 0 and timer 1, both of which are 16-bit.
9. **20 pin I/O Header** – here, data is multiplexed on 16 low-order address bus lines while status lines
10. **Power jack ­– supplies DC current to the 8086 microprocessor kit components.**
11. **Power on switch** – turns the 8086 microprocessor on and off
12. RS 232 -makes communication between the 8086 processor and its peripheral devices as simple as possible by holding the voltage for the data exchange channel between devices and allowing serial data interchange between them.
13. 50 pin bus headers – has some pins for power and many are input and output **pins** for carrying data that can be used to integrate to other cup ports.
14. LCD 16\*2 - has two registers which are command and data that store the command instructions given to the LCD store data to be displayed to LCD respectively.
15. 8251 USART (Universal Synchronous/Asynchronous Receiver/Transmitter) - allows for the communication through serial port of the 8086 microprocessor kit by converting the serial information to parallel.
16. Identify the 8086 microprocessor chip pins shown in Fig. 2 and give their functions.

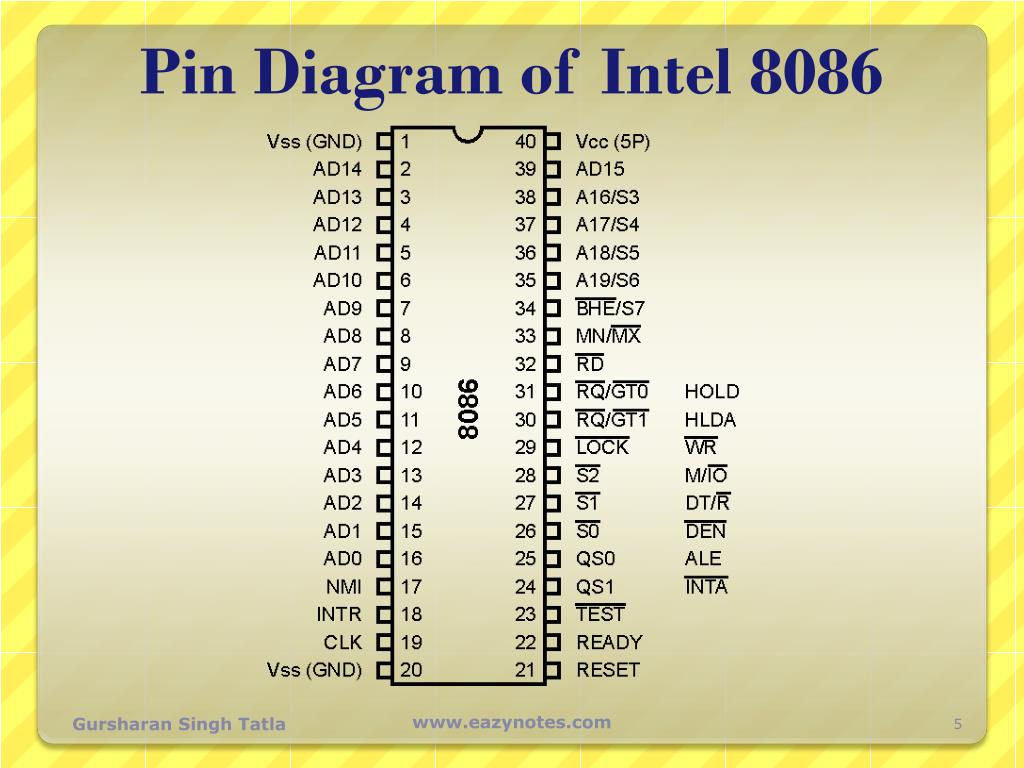


Fig.2 8086 Microprocessor Pin Diagram

* AD0-AD15-The Address/Data Bus are low order address buses that are multiplexed with data. When memory is being transmitted, the symbol A is used instead of D, hence A0-A15 and when data is being transmitted, the symbol D is used hence D0-D15.
* A16-A19-Consists of high order address buses that are multiplexed with status signals.
* S2, S1, S0-Are status pins that are active during T4, T1 and T2 states and are returned to passive state (1, 1, 1 during T3 or TW (when ready is inactive) and are used by the 8228 bus controller for generating all the memory and I/O operation) access control signals. Any change in S2, S1, S0 during T4 indicates the beginning of a bus cycle.
* **VCC** – here, the external power supply of **+ 5V** is provided to the processor.
* **VSS** – These two pins act as the ground by directing the extra current of the microprocessor to ground.
* **BHE’ / S7** – BHE (Bus High Enable) - the combination of the BHE signal and S7 status brings about the existence of the data on the bus and also, the different combinations show whether the bus is containing overall 16 bit, upper byte or lower byte of the data.
* **MN/MX’** – the status here shows whether the processor is operating in the minimum mode or maximum mode by using a signal 0 to show the 8086 operating in maximum mode (multiple processors) and signal 1 to show the 8086 operating in minimum mode (single processor).
* **RD’** – shows the microprocessor performing read operation with either memory or I/O devices with an active low signal.
* **CLK** – a signal here shows the timing to the internal operations that are being executed inside the microprocessor.
* **NMI** – (Non-maskable interrupt). is an uncontrollable interrupt generated inside the processor and when this happens, an interrupt service routine is generated by the interrupt vector table.
* **TEST** – shows the wait instruction, whenever a low signal at this pin occurs then the process inside the processor continues and in the case of the high signal, the processor has to wait for the disabling of this pin.
* **INTR** (interrupt request) – here, the processor after each clock cycle samples the INTR and if the signal is found to be high, then the processor controls that interrupt internally.
* READY - is used by the peripherals and memory devices in order to show the readiness for the next operation.
* RESET – when enabled, it resets the processor and other devices connected to the system by immediately terminating the recent task.
* **NTA’** – is an interrupt acknowledge pin and whenever an INTR signal is generated, then the microprocessor generates INTA signal, as a response to that interrupt.
* **ALE** – (address latch enable). Is enabled by the microprocessor whenever an address is present in the multiplexed address and data bus to inform the peripherals and memory devices about fetching of the data or instruction at that memory location.
* **DEN’** (data enable) - is an active low pin, i.e., whenever a 0 is present, the transceiver gets enabled and separates the data from the multiplexed address and data bus.
* **DT/R’** – shows whether data is getting transmitted or received. A high signal at here provides the information regarding the transmission of data, while a low signal indicates reception of data.
* **M/IO’** – indicates whether the processor is performing an operation with memory or I/O devices and whenever a high is present, it shows the operation is carried out through the memory and whenever there is a low signal, it shows operation through I/O devices.
* **WR’** – here, an active low signal indicates the processor is performing write operation from either memory or I/O devices.
* **HOLD** – When an external device enables this pin, then the processor stops accessing the buses immediately after the recent task gets over.
* **HLDA** – is used as a response pin for the hold request. Once a request for accessing the buses is produced by an external entity, then the microprocessor acknowledges the device that its request will be considered once it gets over by the current operation.
* **LOCK’** – is involved in maximum mode operation, basically meaning that when a single processor is accessing the buses and peripherals, it locks the resources being used by it so that no other entity can access it until the recent processor frees it.
* RQ’/ GT0‘ and RQ’/ GT1‘ –  these pins indicate the requests and granted permissions for accessing buses, memory and peripherals.

**QUESTIONS**

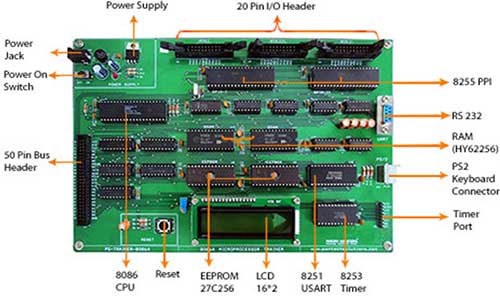
1. Distinguish between the minimum mode and maximum mode of 8086 Microprocessor.

* In minimum mode, there can only be one processor and in this case, it is the 8086 and the circuit is simpler while in maximum mode, there can be multiple processors, 8086, 8085, 8087, etc. and the circuit is more complex. In addition to that, performance in minimum mode is slower and multiprocessing cannot be performed while in maximum mode speed is higher and multiprocessing can be performed

2. Comment on the use of 8086 microprocessor kit and 8086 Emulator.

* An 8086 microprocessor makes operations like multiplication and division easy because it has a powerful instruction set. It has a 16-bit microprocessor, 20-bit address line, 16-bit data bus, 1MB memory capacity, minimum and maximum modes and supports pipelining.
* An 8086 emulator emulates old 8086 processors which were used in Macintosh and Windows computers from the 1980s and early 1990s. Using an 8086 emulator can enable someone to write assembly programs that can run on NECP9801 and early IBM compatible computers.

You may find the following information in Fig.3 and Fig.4. useful.



**END**