


|  |  |   |
|--|--|---|
|  <b>Marwadi University</b><br>Marwadi Chandarana Group | <b>Marwadi University</b><br><b>Department of Computer Engineering</b> |   |
| <b>Subject: Fundamental of Processors (01CE0509)</b>   | <b>Aim:</b>  | <ol style="list-style-type: none"> <li>1. Assembly language programming using MASM/TASM software</li> <li>2. Study of Advance Microprocessor (80286, 80386, 80486, Pentium, ARM)</li> </ol> |
| <b>Experiment No: 14</b>   | <b>Date:</b>   | <b>Enrolment No: 92201703058</b>  |

### **Experiment-14**

#### **INTRODUCTION TO MASM/TASM:**

ASSEMBLY LANGUAGE PROGRAMMING USING MASM SOFTWARE:

This software used to write a program (8086, Pentium processors etc.)

The programs are written using assembly language in editor then compile it. The compiler converts assembly language statements into machine language statements/checks for errors. Then execute the compiled program

There are different softwares developed by different companies for assembly language programming.

They are

MASM -• Microsoft Company.

TASM• - Bore Land Company.

MERITS OF MASM:

1. produces binary code
2. Referring data items by their names rather than by their address.


#### **HOW TO ENTER INTO MASM EDITOR:**

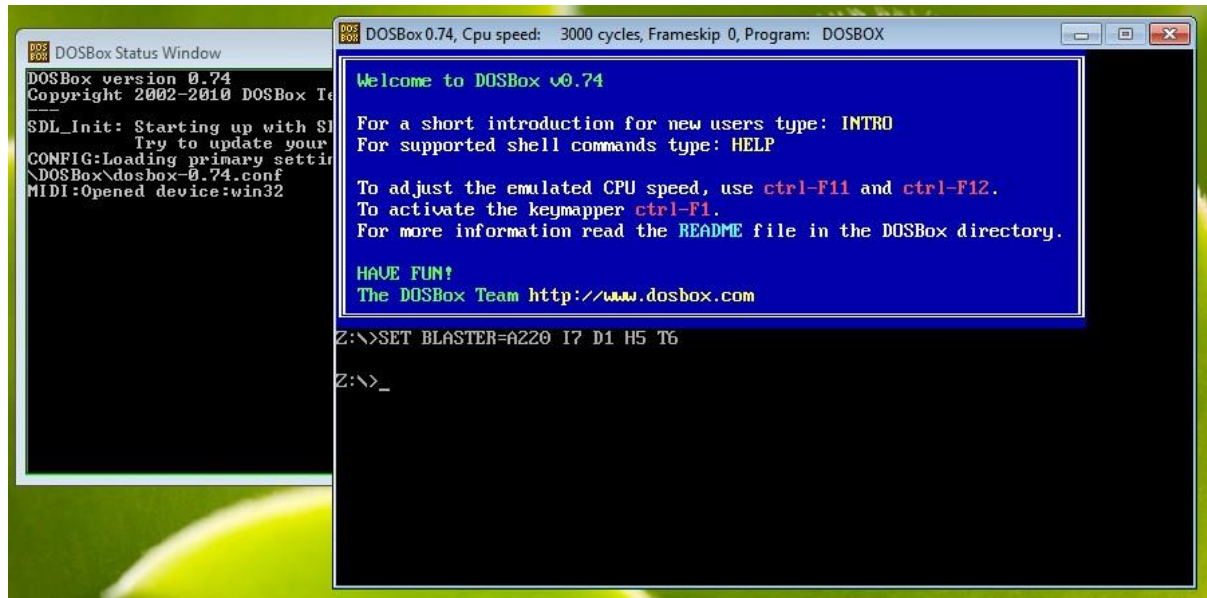
To begin with, download MASM and DosBox software.

1. MASM

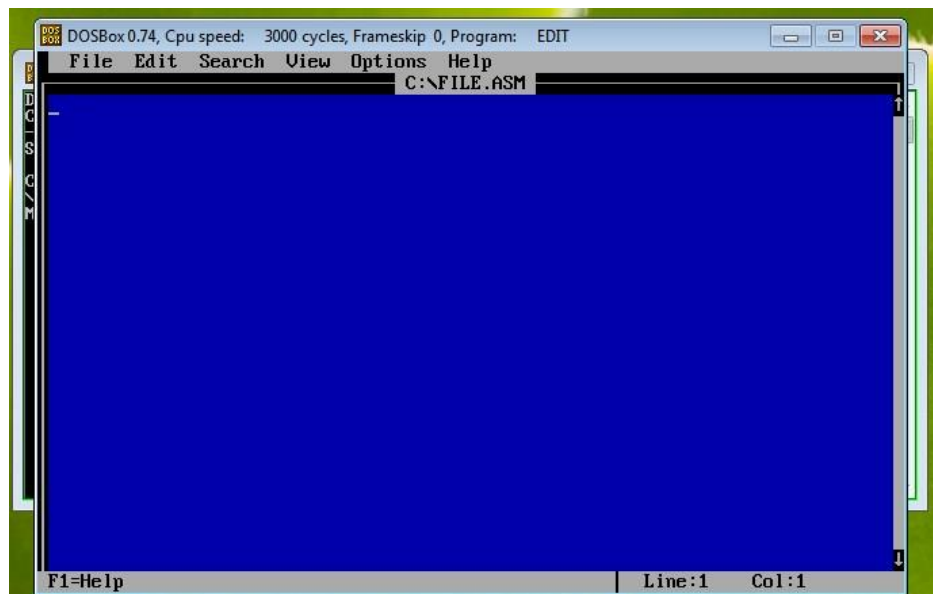
2. DosBox


- After that, extract the MASM file into the 'C' drive, which will create a folder named 8086, then install DosBox.
- Once you have completed the installation, open the DosBox application from your desktop screen. You will see two files:

|  |  |   |
|--|--|---|
|  <b>Marwadi University</b><br>Marwadi Chandarana Group | <b>Marwadi University</b><br><b>Department of Computer Engineering</b> |   |
| <b>Subject: Fundamental of Processors (01CE0509)</b>   | <b>Aim:</b>  | <ol style="list-style-type: none"> <li>1. Assembly language programming using MASM/TASM software</li> <li>2. Study of Advance Microprocessor (80286, 80386, 80486, Pentium, ARM)</li> </ol> |
| <b>Experiment No: 14</b>   | <b>Date:</b>   | <b>Enrolment No: 92201703058</b>  |



- Now, type 'mount c c:\8086' and press Enter,
- followed by 'c:' and press Enter again.
- You are now ready to write your first program.
- Open an editor by typing 'EDIT program\_name.ASM', and a new window will appear as shown below where you can write your program:

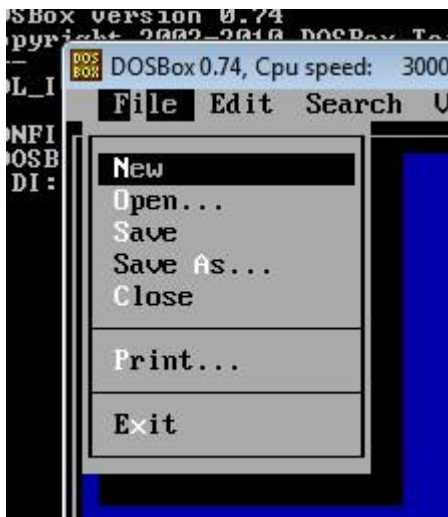


|  |  |   |
|--|--|---|
|  <b>Marwadi University</b><br>Marwadi Chandarana Group | <b>Marwadi University</b><br><b>Department of Computer Engineering</b> |   |
| <b>Subject: Fundamental of Processors (01CE0509)</b>   | <b>Aim:</b>  | <ol style="list-style-type: none"> <li>1. Assembly language programming using MASM/TASM software</li> <li>2. Study of Advance Microprocessor (80286, 80386, 80486, Pentium, ARM)</li> </ol> |
| <b>Experiment No: 14</b>   | <b>Date:</b>   | <b>Enrolment No: 92201703058</b>  |

For example,

### Program:


You need to save the file with an extension '.asm', such as 'hello.asm.'



- Once you have saved the program, exit the editor, and you will be returned to the previous DOSBox window. You should see 'C:>'.
- Now you will write the following command: **masm program-name.asm**
- In our example, I will use the following command: **masm hello.asm**
- Then press Enter. You will see the following:

```
C:\>masm HELLO.ASM
Microsoft (R) Macro Assembler Version 5.00
Copyright (C) Microsoft Corp 1981-1985, 1987.

Object filename [HELLO.OBJ]:
```

|  |  |   |
|--|--|---|
|  <b>Marwadi University</b><br>Marwadi Chandarana Group | <b>Marwadi University</b><br><b>Department of Computer Engineering</b> |   |
| <b>Subject: Fundamental of Processors (01CE0509)</b>   | <b>Aim:</b>  | <ol style="list-style-type: none"> <li>1. Assembly language programming using MASM/TASM software</li> <li>2. Study of Advance Microprocessor (80286, 80386, 80486, Pentium, ARM)</li> </ol> |
| <b>Experiment No: 14</b>   | <b>Date:</b>   | <b>Enrolment No: 92201703058</b>  |

- Now, press Enter again. You will see the following:

```
Object filename [HELLO.OBJ]:
Source listing [NUL.LST]:
```

- Press Enter again. You will see the following:

```
C:\>masm HELLO.ASM
Microsoft (R) Macro Assembler Version 5.0
Copyright (C) Microsoft Corp 1981-1985, 1996-1998

Object filename [HELLO.OBJ]:
Source listing [NUL.LST]:
Cross-reference [NUL.CRF]:

51756 + 464788 Bytes symbol space free

0 Warning Errors
0 Severe Errors



C:\>
```

- If there are any errors or warnings, you will be notified of them, and you need to correct them by typing 'EDIT program\_name.ASM.'
- Once you have fixed the issues, type '**link program\_name\_without\_extension**'. In our example, we use the following: **LINK hello**
- Press Enter. You will see the following:

```
C:\>link HELLO

Microsoft (R) Overlay Linker Version 3.60
Copyright (C) Microsoft Corp 1983-1987. All rights reserved.

Run File [HELLO.EXE]:
List File [NUL.MAP]:
Libraries [.LIB]:
LINK : warning L4021: no stack segment
```

|   |  |                                  |
|---|--|----------------------------------|
|  <b>Marwadi University</b><br>Marwadi Chandarana Group  | <b>Marwadi University</b><br><b>Department of Computer Engineering</b>   |                                  |
| <b>Subject: Fundamental of Processors (01CE0509)</b>  | <b>Aim:</b><br>1. Assembly language programming using MASM/TASM software<br>2. Study of Advance Microprocessor (80286, 80386, 80486, Pentium, ARM) |                                  |
| <b>Experiment No: 14</b>  | <b>Date:</b>   | <b>Enrolment No: 92201703058</b> |

Finally, type 'program-name.exe' .For our example, it would be: **hello.exe**

### Output:

To exit the DOSBox window, type 'exit'.

### Addition of Two 8-bit Numbers Using Assembly Language in TASM:

#### Objective:

To write an assembly language program that adds two 8-bit numbers using TASM in DOSBox.

#### Apparatus/Software Required:


1. A PC with Windows (64-bit or 32-bit)
2. **TASM** (Turbo Assembler) installed in DOSBox
3. DOSBox emulator for running 16-bit programs

#### Theory:

Assembly language is a low-level programming language that directly interacts with the hardware. It is useful for gaining an understanding of how a microprocessor functions. In this experiment, we use the **8086 microprocessor**, where basic arithmetic operations such as addition are performed using registers like **AL** and **AH**.

#### Program Description:

- We will use the **MOV** instruction to load values into registers.
- The **ADD** instruction adds two numbers.
- The result is stored in the **AX** register (8-bit numbers stored in the AL register).

|  |  |                                  |
|--|--|----------------------------------|
|  <b>Marwadi University</b><br>Marwadi Chandarana Group | <b>Marwadi University</b><br><b>Department of Computer Engineering</b>   |                                  |
| <b>Subject: Fundamental of Processors (01CE0509)</b>   | <b>Aim:</b><br>1. Assembly language programming using MASM/TASM software<br>2. Study of Advance Microprocessor (80286, 80386, 80486, Pentium, ARM) |                                  |
| <b>Experiment No: 14</b>   | <b>Date:</b>   | <b>Enrolment No: 92201703058</b> |

### Algorithm:

1. Load the first 8-bit number into the **AL** register.
2. Load the second 8-bit number into the **BL** register.
3. Add the contents of **AL** and **BL** using the **ADD** instruction.
4. Store the result in memory.

### Assembly Program:

### Step-by-Step Procedure:

#### 1. Install TASM and DOSBox:



- Download and install **DOSBox** and place the **TASM** folder in the root directory (C:).
- Mount the TASM folder in DOSBox by typing: **mount c c:\TASM**
- This mounts the TASM directory.

#### 2. Write the Program:

- Open DOSBox and navigate to the TASM folder. Type: **c:**
- Use the **edit** command to write the assembly program: **edit add.asm**
- Type the above code into the editor and save it.

#### 3. Assemble the Program:

- After writing the program, assemble it using **TASM**: **tasm add.asm**

|  |  |   |
|--|--|---|
|  <b>Marwadi University</b><br><small>Marwadi Chandarana Group</small>  | <b>Marwadi University</b><br><b>Department of Computer Engineering</b> |   |
| <b>Subject: Fundamental of Processors (01CE0509)</b>   | <b>Aim:</b>  | <ol style="list-style-type: none"> <li>1. Assembly language programming using MASM/TASM software</li> <li>2. Study of Advance Microprocessor (80286, 80386, 80486, Pentium, ARM)</li> </ol> |
| <b>Experiment No: 14</b>   | <b>Date:</b>   | <b>Enrolment No: 92201703058</b>  |

- This will produce an object file (add.obj).

#### 4. Link the Program:

- Link the object file using **TLINK: tlink add.obj**

#### 5. Run the Program:


- After linking, run the program: **td add.exe**

**TASM assembler before Execution:**

**TASM assembler final output:**

For Single step debugging Press F7 to execute single instruction and check the result after every step.

- The result of adding the two 8-bit numbers (5H + 3H = 8H).

|  |  |                                  |
|--|--|----------------------------------|
|  <b>Marwadi University</b><br>Marwadi Chandarana Group | <b>Marwadi University</b><br><b>Department of Computer Engineering</b>   |                                  |
| <b>Subject: Fundamental of Processors (01CE0509)</b>   | <b>Aim:</b><br>1. Assembly language programming using MASM/TASM software<br>2. Study of Advance Microprocessor (80286, 80386, 80486, Pentium, ARM) |                                  |
| <b>Experiment No: 14</b>   | <b>Date:</b>   | <b>Enrolment No: 92201703058</b> |

### **Study of Advance Microprocessor (80286, 80386, 80486, Pentium, ARM):**

#### **1. 80286 Features**

The Intel 80286, commonly referred to as the 286, is a microprocessor that was released by Intel in 1982. It represents an important step in the evolution of the x86 architecture and was widely used in personal computers during the 1980s, particularly in IBM's PC/AT and compatible systems.

#### **Key Features and Specifications**

##### **1. Architecture:**

- The 80286 is a 16-bit microprocessor, meaning it can process data in 16-bit chunks.
- It introduced significant enhancements over its predecessor, the 8086/8088, particularly in memory management and multitasking.

##### **2. Clock Speed:**

- The 80286 was available in a range of clock speeds, from 6 MHz to 25 MHz. This was a considerable improvement over the 8086, which operated at up to 10 MHz.

##### **3. Transistor Count:**

- The 80286 contains approximately 134,000 transistors, which was a substantial increase from the 29,000 transistors in the 8086.


##### **4. Memory Addressing:**

- One of the most significant advancements of the 80286 was its ability to address up to 16 MB of RAM using a 24-bit address bus. This was a major improvement over the 1 MB limit of the 8086.
- It introduced a new *protected mode*, which allowed the processor to access and manage this larger memory space more effectively. This mode also provided hardware-based memory protection, which helped prevent programs from interfering with each other's memory space.

##### **5. Protected Mode vs. Real Mode:**

- **Real Mode:** In real mode, the 80286 behaves similarly to the 8086, with access to only 1 MB of memory and no memory protection. This mode was primarily used for compatibility with older software.
- **Protected Mode:** In protected mode, the 80286 could manage larger memory spaces and offer better multitasking and memory protection. However, one limitation was that the 80286 could not easily switch back to real mode from



|  |  |                                  |
|--|--|----------------------------------|
|  <b>Marwadi University</b><br>Marwadi Chandarana Group | <b>Marwadi University</b><br><b>Department of Computer Engineering</b>   |                                  |
| <b>Subject: Fundamental of Processors (01CE0509)</b>   | <b>Aim:</b><br>1. Assembly language programming using MASM/TASM software<br>2. Study of Advance Microprocessor (80286, 80386, 80486, Pentium, ARM) |                                  |
| <b>Experiment No: 14</b>   | <b>Date:</b>   | <b>Enrolment No: 92201703058</b> |

protected mode without a reset, which was a drawback for running DOS-based applications.

#### 6. Multitasking:

- The 80286 introduced support for basic multitasking, allowing the operating system to switch between multiple programs. However, this feature was not widely utilized in consumer operating systems of the time, as DOS and early versions of Windows primarily operated in real mode.

#### 7. Instruction Set:

- The 80286 supports a subset of the x86 instruction set, with additional instructions to support the new protected mode features. It is compatible with software written for the 8086/8088 processors.

## 2. 80386 Features

The Intel 80386, often referred to as the 386, was a significant advancement in the history of microprocessors and the evolution of the x86 architecture. Introduced by Intel in 1985, the 80386 was the first 32-bit processor in the x86 family, bringing substantial improvements in performance, memory management, and capabilities over its predecessor, the 80286.

### Key Features and Specifications


#### 1. Architecture:

- **32-bit Processor:** The 80386 was Intel's first 32-bit processor, meaning it could handle 32-bit data and had a 32-bit wide data bus, allowing for faster and more efficient data processing.
- **32-bit Address Bus:** With a 32-bit address bus, the 80386 could address up to 4 GB of RAM, a huge leap from the 16 MB limit of the 80286. This capability was crucial as software and systems began to demand more memory.

#### 2. Clock Speed:

- The 80386 was available in various clock speeds, typically ranging from 12 MHz to 40 MHz. Higher clock speeds contributed to the processor's enhanced performance.

#### 3. Transistor Count:

|  |  |                                  |
|--|--|----------------------------------|
|  <b>Marwadi University</b><br>Marwadi Chandarana Group | <b>Marwadi University</b><br><b>Department of Computer Engineering</b>   |                                  |
| <b>Subject: Fundamental of Processors (01CE0509)</b>   | <b>Aim:</b><br>1. Assembly language programming using MASM/TASM software<br>2. Study of Advance Microprocessor (80286, 80386, 80486, Pentium, ARM) |                                  |
| <b>Experiment No: 14</b>   | <b>Date:</b>   | <b>Enrolment No: 92201703058</b> |

- The 80386 contained approximately 275,000 transistors, a significant increase from the 134,000 transistors in the 80286, reflecting the increased complexity and capabilities of the chip.

#### 4. Operating Modes:

- **Real Mode:** The 80386 could operate in real mode, which was backward compatible with the 16-bit 8086/8088 and 80286 processors. In real mode, it could address only 1 MB of memory, making it suitable for running older software.
- **Protected Mode:** Like the 80286, the 80386 supported protected mode, but with full 32-bit capabilities. In protected mode, the 80386 could utilize its full 4 GB address space and offer advanced memory protection and multitasking features, laying the groundwork for modern operating systems.
- **Virtual 8086 Mode:** A new feature introduced with the 80386, Virtual 8086 mode allowed the processor to run multiple 16-bit real-mode applications in isolated, protected environments, effectively enabling multitasking of legacy software under modern operating systems.

#### 5. Paging and Virtual Memory:


- The 80386 introduced hardware support for paging, a critical feature for implementing virtual memory. This allowed the operating system to use disk space as an extension of RAM, providing more flexibility in memory management and enabling larger programs to run on systems with limited physical memory.

#### 6. Instruction Set:

- The 80386 extended the x86 instruction set to support 32-bit operations, adding new instructions and capabilities while maintaining backward compatibility with 16-bit software. This ensured that existing programs could run on the new processor without modification.

### 3. 80486 Features

The Intel 80486, commonly known as the 486, is a significant processor in the evolution of the x86 architecture. Introduced in 1989, the 80486 built upon the foundation laid by the 80386 but added several key enhancements that made it a more powerful and efficient processor. It was widely used in personal computers during the early to mid-1990s and played a crucial role in the development of modern computing.

|  |  |                                  |
|--|--|----------------------------------|
|  <b>Marwadi University</b><br>Marwadi Chandarana Group | <b>Marwadi University</b><br><b>Department of Computer Engineering</b>   |                                  |
| <b>Subject: Fundamental of Processors (01CE0509)</b>   | <b>Aim:</b><br>1. <b>Assembly language programming using MASM/TASM software</b><br>2. <b>Study of Advance Microprocessor (80286, 80386, 80486, Pentium, ARM)</b> |                                  |
| <b>Experiment No: 14</b>   | <b>Date:</b>   | <b>Enrolment No: 92201703058</b> |

#### Key Features and Specifications

- **Integrated Floating Point Unit (FPU):** One of the major advancements in the 80486 was the integration of the Floating Point Unit (FPU) directly onto the processor die. This was a separate co-processor in earlier systems (such as the 80387 for the 80386) and was responsible for handling complex mathematical calculations, particularly in scientific and engineering applications.
- **Integrated Level 1 (L1) cache (8 KB),** which significantly boosted performance.
- **Introduced power-saving features** with the 486SL variant, making it suitable for portable devices.

#### 4. Pentium Features

The Intel Pentium, often referred to as the Pentium 1, was a major milestone in the development of personal computing. Released in 1993, the Pentium was the first of Intel's x86 processors to be branded under a name rather than a number. It introduced several key innovations that significantly improved processing power, efficiency, and performance, making it a foundational technology for modern personal computers.

#### Key Features and Specifications



##### 1. Architecture:

- **32-bit Processor:** Like its predecessors, the Pentium is a 32-bit processor, but it introduced a more advanced and efficient design compared to the earlier 80486.
- **Superscalar Architecture:** One of the most significant advancements in the Pentium was its superscalar architecture, which allowed the processor to execute multiple instructions per clock cycle. This was achieved by incorporating two parallel instruction pipelines, known as the U-pipe and V-pipe. This parallelism enabled the Pentium to perform more operations simultaneously, greatly enhancing performance.

##### 2. Clock Speed:

- The Pentium was released with clock speeds starting at 60 MHz and 66 MHz. Later models, like the Pentium 75 and beyond, increased clock speeds up to 300 MHz, offering significant performance improvements over earlier processors.

##### 3. Transistor Count:

|   |  |                                  |
|---|--|----------------------------------|
|  <b>Marwadi University</b><br>Marwadi Chandarana Group  | <b>Marwadi University</b><br><b>Department of Computer Engineering</b>   |                                  |
| <b>Subject: Fundamental of Processors (01CE0509)</b>  | <b>Aim:</b><br><ol style="list-style-type: none"> <li>1. Assembly language programming using MASM/TASM software</li> <li>2. Study of Advance Microprocessor (80286, 80386, 80486, Pentium, ARM)</li> </ol> |                                  |
| <b>Experiment No: 14</b>  | <b>Date:</b>   | <b>Enrolment No: 92201703058</b> |

- The Pentium contained approximately 3.1 million transistors in its original versions, a significant increase over the 1.2 million transistors in the 80486. This increase in transistor count allowed for more complex circuits, greater functionality, and improved performance.

#### 4. Cache Memory:

- **L1 Cache:** The Pentium featured a split Level 1 (L1) cache, with separate 8 KB caches for data and instructions. This split cache design helped reduce memory access times and improved overall performance.
- **L2 Cache (External):** While the Pentium did not have an on-chip L2 cache, it supported an external L2 cache, typically ranging from 256 KB to 512 KB, which further improved system performance.

#### 5. Branch Prediction:

- The Pentium introduced an advanced branch prediction unit, which helped improve the efficiency of executing conditional instructions (branches). By predicting the direction of branches (whether a condition would be true or false), the processor could prefetch and execute instructions more effectively, reducing delays caused by incorrect predictions.

#### 6. Floating Point Unit (FPU):


- The Pentium included an integrated Floating Point Unit (FPU), which was significantly more powerful than those in earlier processors. This improved performance in applications that required heavy mathematical computations, such as scientific simulations, 3D graphics, and engineering software.

#### 7. Instruction Set:

- The Pentium extended the x86 instruction set, maintaining backward compatibility with earlier processors while introducing new instructions to optimize performance.

#### 8. Multimedia Extensions (MMX):

- Later versions of the Pentium, known as Pentium MMX, introduced MMX technology, which added 57 new instructions specifically designed to enhance multimedia processing. MMX improved the performance of audio, video, and

|  |  |                                  |
|--|--|----------------------------------|
|  <b>Marwadi University</b><br>Marwadi Chandarana Group | <b>Marwadi University</b><br><b>Department of Computer Engineering</b>   |                                  |
| <b>Subject: Fundamental of Processors (01CE0509)</b>   | <b>Aim:</b><br><ol style="list-style-type: none"> <li>1. Assembly language programming using MASM/TASM software</li> <li>2. Study of Advance Microprocessor (80286, 80386, 80486, Pentium, ARM)</li> </ol> |                                  |
| <b>Experiment No: 14</b>   | <b>Date:</b>   | <b>Enrolment No: 92201703058</b> |

graphics applications by enabling the processor to handle multiple data elements in parallel.

## 6. ARM Processor

ARM processors, based on the ARM (Advanced RISC Machine) architecture, are a family of CPUs that are widely used in a variety of devices, from smartphones and tablets to embedded systems, servers, and even some laptops. ARM processors are known for their energy efficiency, scalability, and flexibility, making them a popular choice for many applications where power consumption and cost are critical factors.

### Key Features and Characteristics

#### 1. RISC Architecture:

- ARM processors are built on the **RISC (Reduced Instruction Set Computing)** architecture, which is designed to perform a smaller number of types of computer instructions so that they can be executed very quickly. This contrasts with CISC (Complex Instruction Set Computing) architectures like Intel's x86, which use a broader set of instructions.
- The simplicity of the RISC design allows for higher performance per watt, making ARM processors more power-efficient compared to many CISC processors.


#### 2. Energy Efficiency:

- One of the defining characteristics of ARM processors is their low power consumption. This makes them ideal for battery-powered devices like smartphones, tablets, and wearables, where energy efficiency is crucial for extending battery life.

#### 3. Scalability:

- ARM processors are highly scalable, ranging from simple microcontrollers used in embedded systems to powerful multi-core processors used in servers and high-end mobile devices.
- The ARM architecture can be customized and licensed to different manufacturers, allowing companies to tailor ARM cores to their specific needs. This has led to a wide variety of ARM-based chips with different capabilities and performance levels.

#### 4. Instruction Set:

|  |  |                                  |
|--|--|----------------------------------|
|  <b>Marwadi University</b><br>Marwadi Chandarana Group | <b>Marwadi University</b><br><b>Department of Computer Engineering</b>   |                                  |
| <b>Subject: Fundamental of Processors (01CE0509)</b>   | <b>Aim:</b><br><ol style="list-style-type: none"> <li>1. Assembly language programming using MASM/TASM software</li> <li>2. Study of Advance Microprocessor (80286, 80386, 80486, Pentium, ARM)</li> </ol> |                                  |
| <b>Experiment No: 14</b>   | <b>Date:</b>   | <b>Enrolment No: 92201703058</b> |

- The ARM architecture features a simplified and efficient instruction set, which includes a fixed-length 32-bit instruction set (and 16-bit in some cases with the Thumb instruction set for greater efficiency).
- ARM processors also support various execution modes and features like conditional execution, load/store multiple, and powerful branch prediction.

#### 5. License-Based Model:

- ARM Ltd. does not manufacture processors itself but instead licenses its architecture and designs to other companies, like Qualcomm, Apple, Samsung, and NVIDIA, who then design and produce their own ARM-based chips. This licensing model has allowed ARM to become ubiquitous across different types of devices.

#### 6. Multi-Core and 64-bit Support:

- Modern ARM processors support multi-core configurations, allowing for increased parallelism and performance.
- ARM introduced 64-bit processing with the ARMv8 architecture, which added support for larger address spaces and more powerful computational capabilities, making ARM processors suitable for more demanding applications, including servers and high-performance computing.

#### 7. Security Features:

- ARM processors include security features like ARM TrustZone, which creates a secure execution environment for running sensitive code, helping to protect against malware and other security threats.

#### Conclusion: