# AKASH M

Email: akashm.eee.rymec@gmail.com | Phone Number: 6362342678

LinkedIn: www.linkedin.com/in/akash-m-364b613537

## CAREER OBJECTIVE

To pursue a challenging engineering role where I can apply my programming and embedded systems knowledge, continuously learn new technologies, and grow into a competent professional by contributing to practical, real-world solutions.

## WORK EXPERIENCE

- Currently undergoing technical training program – **Advanced Embedded Systems Course** at Emertxe Information Technologies (http://www.emertxe.com) Bangalore
- This course is Government of India certified program, aligned with **Skill India** / NSDC under Electronics Sector Skill Council of India (http://www.essc-india.org) - **Embedded Software Engineer QP ELE /Q1501**

## TECHNICAL SKILLS

- Programming Languages:
  - Shell scripting
  - Advanced C programming
  - OOP using C++
  - Data structures
- System programming:
  - Linux Kernel system calls
  - IPC mechanisms – Pipe, FIFO, Shared memory
  - Network Programming using TCP and UDP sockets
  - pThreads – Multi thread programming
- Development environment and tools:
  - Dev environment: Vim, Makefiles
  - Compilers: GCC

## COURSE WORK

- Microprocessor
- Digital Electronics
- Digital Signal Processing
- Embedded systems Fundamentals
- C/C++ Programming

## EDUCATION

- B.E (EEE) | VTU | 7.63 | 2022-2025
- DIPLOMA(EEE) | DTE | 69.10% | 2019-2022
- Class – X | CBSE | 53% | 2018-2019

---

## PROJECTS

**Project Number:1**

| | |
|---|---|
| **Title** | **Image Steganography using LSB Encoding and Decoding** |
| **Project brief** | Built an application to securely hide secret text data within image files using steganography techniques, ensuring minimal visual distortion and data confidentiality. |
| **Technologies used** | Advanced C – File handling, Pointers, Bitwise operations, Structures, Command-line arguments. |
| **Key challenges & Learnings** | ✓ Embedding data into image files without noticeable visual changes. Learned practical implementation of Least Significant Bit (LSB) techniques Implementing precise bitwise manipulation to avoid image distortion.<br>✓ Correctly extracting hidden data without loss or corruption. Improved accuracy in bitwise operations and data extraction logic<br>✓ Balancing data security and image quality. Gained foundational understanding of data security concepts trade-offs. |

**Project number:2**

| | |
|---|---|
| **Title** | **Address Book Management System** |
| **Project brief** | Designed and developed a menu-driven console application to manage contact information efficiently. The system supports creating, searching, updating, deleting, and displaying contact records with permanent storage. |
| **Technologies used** | Advanced C, Structures, File Handling |
| **Key challenges & Learnings** | ✓ Gained hands-on experience in using structures to organize contact information.<br>✓ Learned file handling techniques to store and retrieve data permanently.<br>✓ Improved understanding of pointers and memory management.<br>✓ Enhanced logical thinking by implementing menu-driven operations and validations. |

**Project number:3**

| | |
|---|---|
| **Title** | **MP3 Tag Reader and editor (ID3V2)** |
| **Project brief** | This project involved developing an MP3 tag reader to extract metadata such as **song title, artist, album, year, and genre** from MP3 files by parsing **ID3 tags**. The application reads binary MP3 data, identifies tag headers, and displays meaningful information to the user. |
| **Technologies used** | Advanced C – File I/O, Structures, Bitwise operations, String manipulation, Pointers |

| | |
|---|---|
| **Key challenges & Learnings** | ✓ Understanding the internal structure of MP3 and ID3 tag formats. Gained in-depth knowledge of multimedia file formats and headers<br>✓ Parsing binary files accurately without data misalignment. Strengthened skills in binary file handling and byte-level data manipulation<br>✓ Extracting meaningful information from raw binary data. Improved understanding of memory layout and low-level file processing |

**Project number:4**

| **Title** | **Lexical Analyzer** |
|---|---|
| **Project brief** | Developed a Lexical Analyzer to convert a stream of characters from a C source file into a sequence of lexical tokens. The program reads the input C file and identifies tokens such as keywords, identifiers, operators, literals, and punctuation symbols. The generated tokens can be used as input for the next phase of the compilation process. The project also includes error handling for invalid or unrecognized tokens during tokenization. |
| **Technologies used** | Advanced C, Function pointers, String operations, File I/O handling, Conditional statements |
| **Key challenges & Learnings** | ✓ Gained understanding of lexical analysis and tokenization process.<br>✓ Learned how to identify different token types in C language.<br>✓ Implemented error handling for invalid characters and syntax.<br>✓ Improved logical thinking and pattern-matching skills. |

**Project number:5**

| **Title** | **Mini Shell** |
|---|---|
| **Project brief** | Developed a minimalistic Linux shell named **msh** to execute user commands through a command-line interface. The shell provides a customizable prompt, executes external commands using process creation, and supports built-in commands. It also handles signals, background job control, special shell variables, and inter-process communication using pipes, closely simulating basic functionality of a Linux shell. |
| **Technologies used** | Advanced C, Linux system calls (fork, exec, wait, pipe), Signal handling, Process synchronization, File and string operations. |
| **Key challenges & Learnings** | ✓ Gained deep understanding of process creation and management in Linux.<br>✓ Learned signal handling and job control mechanisms.<br>✓ Improved knowledge of command parsing and inter-process communication using pipes.<br>✓ Enhanced debugging and system-level programming skills. |

**Project number:6**

| Title | TFTP Server and client |
|---|---|
| Project brief | The Trivial File Transfer Protocol (TFTP) is a simple protocol used to transfer files between two systems over a network. It is defined in RFC 1350. Unlike FTP, TFTP does not support advanced features such as user authentication and complex commands. Because of its simplicity and small size, it is commonly used in embedded systems. In this project, the TFTP protocol was implemented in a Local Area Network (LAN) to transfer files between client and server systems. The project helps understand how network protocols work and how file transfer happens using UDP communication. |
| Technologies used | Advanced C, TCP/IP Networking, UDP Socket APIs, Timeout and Retransmission Mechanism, Linux System Programming. |
| Key challenges & Learnings | ✓ Learned how **network communication works using UDP sockets**.<br>✓ Gained understanding of **network protocols and RFC standards**.<br>✓ Implemented **timeout and retransmission mechanisms for reliable communication**.<br>✓ Improved knowledge of **socket programming and file transfer in networks**.<br>✓ Enhanced **debugging and system-level programming skills**. |

**Project number:7**

| Title | Arbitrary Precision Calculator (APC) |
|---|---|
| Project brief | Arbitrary Precision Calculator is a program that performs arithmetic operations on very large numbers that cannot be handled by normal data types like int or long. In this project, large numbers are stored using linked lists and mathematical operations are performed on them. The calculator supports operations such as addition, subtraction, multiplication, modulus, and power between two large numbers. This project helps understand how big number calculations work using data structures. |
| Technologies used | Advanced C, Linked List Data Structure, Abstract Data Types (ADT), Algorithm Complexity (Time Complexity), Dynamic Memory Allocation. |
| Key challenges & Learnings | ✓ Gained deep understanding of process creation and management in Linux.<br>✓ Learned signal handling and job control mechanisms.<br>✓ Learned how to **handle very large numbers using linked lists**.<br>✓ Improved understanding of **Abstract Data Types (ADT)**.<br>✓ Implemented **mathematical operations on large numbers**.<br>✓ Gained knowledge of **algorithm complexity and optimization**.<br>✓ Enhanced problem-solving and debugging skills. |

**Project number:8**

| | |
|---|---|
| **Title** | **Inverted Search** |
| **Project brief** | Inverted Search is a program that searches words from multiple text files efficiently. The program reads several .txt files, processes each word, and stores them in a hash table using linked list data structures. Each word is stored along with the file name and the number of times it appears in that file. The system allows the user to create the database, display stored data, search for a word, save the database to a file, and update the database from a saved file. This project demonstrates fast searching using hashing and linked lists. |
| **Technologies used** | Advanced C, Hash Table, Linked Lists, File Handling, String Manipulation, Data Structures |
| **Key challenges & Learnings** | ✓ Gained deep understanding of process creation and management in Linux.<br>✓ Learned signal handling and job control mechanisms.<br>✓ Learned how to implement **hash tables for fast searching**.<br>✓ Improved understanding of **linked list data structures**.<br>✓ Gained experience in **handling multiple files and word parsing**.<br>✓ Implemented **database creation, search, save, and update operations**.<br>✓ Enhanced debugging and problem-solving skills in C programming. |

## ACADEMIC PROJECTS

| | |
|---|---|
| **Title** | **Voice Assistant With CHATGPT** |
| **Project brief** | The **Voice Assistant Using ChatGPT** project enables users to interact with a system using voice commands. It converts speech into text, processes the input using ChatGPT, and generates intelligent responses. The reply is then converted into speech for natural interaction. This project demonstrates the use of AI and natural language processing in real-time applications. |
| **Technologies used** | Python, Speech Recognition, Raspberry PI, ChatGPT by OpenAI, pyttsx3 / Google Text-to-Speech, Microphone and Speaker |
| **Key challenges & Learnings** | ✓ One of the major challenges was accurately converting voice input into text in noisy environments.<br>✓ Integrating the ChatGPT API and handling real-time responses required a clear understanding of API communication and error handling.<br>✓ Managing response speed and ensuring natural-sounding voice output was also challenging.<br>✓ Through this project, we learned about speech processing, natural language understanding, and real-time AI system integration. |

| | |
|---|---|
| **Title** | **PIR Motion Sensor Detector** |
| **Project brief** | The **PIR Motion Sensor Detector** project is used to detect human movement by sensing changes in infrared radiation. When motion is detected, the sensor activates an output such as an alarm or light. This system is widely used in security and automatic lighting applications. It is simple, low-cost, and energy efficient. |
| **Technologies used** | HC-SR 501 PIR Motion Sensor module, Buzzer, LED |
| **Key challenges & Learnings** | ✓ One of the main challenges was accurately detecting human motion while minimizing false triggers caused by heat changes or environmental noise.<br>✓ Interfacing the PIR sensor with the microcontroller and calibrating its sensitivity and delay time required careful testing.<br>✓ Through this project, we learned about infrared sensing principles, microcontroller interfacing, and real-time signal processing.<br>✓ The project also improved our understanding of embedded systems and practical hardware implementation. |