AMS-DOS

The Advanced Microcontroller System - DOS for Arduino

AMS-DOS is a lightweight, efficient operating system designed specifically for Arduino microcontrollers. Built to maximize the potential of limited resources, AMS-DOS provides a structured environment for managing tasks, memory, and peripherals. This OS enables Arduino enthusiasts and developers to organize and execute complex programs on simple hardware, bringing multi-tasking and a more structured development approach to the Arduino platform.

1. Key Features

1. **Basic Task Scheduler**
    - Supports cooperative multitasking, allowing multiple tasks to run in sequence without interruption.
    - Provides delay and timing functions to schedule task execution.
2. **Memory Management**
    - Allocates and deallocates memory dynamically to optimize limited SRAM usage.
    - Includes memory monitoring tools to help developers track and manage memory use efficiently.
3. **File System Support**
    - Supports basic file management on connected SD cards, allowing file creation, reading, writing, and deletion.
    - Enables the storage and retrieval of data for persistent applications.
4. **Peripheral Management**
    - Provides built-in drivers for common peripherals (e.g., UART, I2C, SPI, and GPIO management).
    - Simplifies interfacing with connected devices like sensors, displays, and communication modules.
5. **Command Line Interface (CLI)**
    - Offers a simple CLI that can be accessed over serial communication for debugging and direct control of the Arduino.
    - Commands include system info, memory status, task listing, and file management.
6. **Real-Time Clock (RTC) Integration**
    - Supports integration with RTC modules for time-based tasks and data logging.
    - Allows scheduling of tasks based on real-world time.
7. **Inter-Task Communication**
    - Provides basic message-passing functionality for tasks to communicate and synchronize with each other.
8. **Debugging Tools**

    - Includes debugging functions like logging, error tracking, and system diagnostics over serial output.
    - Helps developers monitor task execution, performance, and system status.

9. **Power Management**

    - Optimizes power usage by enabling sleep modes and power-saving features, extending battery life for mobile applications.
    - Manages CPU cycles to reduce power usage during low-demand periods.

10. **Basic Networking Support**

    - Compatible with Wi-Fi and Bluetooth modules (such as ESP8266, ESP32) for basic networking tasks.
    - Enables sending and receiving data over networks for IoT applications.

2. Benefits

AMS-DOS is built for developers who want more control and efficiency in their Arduino projects. By providing the essential features of an operating system, AMS-DOS allows for more structured and scalable programs, making Arduino applications more robust, reliable, and ready for real-world deployment. Whether it's a complex robotics system or a simple sensor logger, AMS-DOS brings a new level of organization and capability to Arduino projects.

3. Modules

1. **Task Scheduler** - `taskmgmt`
2. **Memory Management** - `memmgmt`
3. **File System Support** - `filesys`
4. **Peripheral Management** - `periphmg`
5. **Command Line Interface (CLI)** - `serialio`
6. **RTC Integration** - `rtcclock`
7. **Inter-Task Communication** - `msgpass`
8. **Debugging Tools** - `debugger`
9. **Power Management** - `powermgt`
10. **Networking Support** - `netcomms`

4. Usage

In the AMS-DOS repository, each module is organized into its dedicated folder. This structure makes the codebase easy to navigate, modular, and scalable. Each folder contains the main `.ino` file along with any necessary `.h` (header) and `.cpp` (source) files for that module's functionality.

Here's how each module is structured:

1. **Folder Structure**: Each module resides in a folder named after its respective 8-character identifier (e.g., `taskmgmt`, `memmgmt`, etc.).

2. **Module Contents**:

- `.ino` **File**: Contains the main code for the module. This file typically serves as the entry point for the module's functionality and interacts with the primary AMS-DOS system.
- `.h` **File**: The header file declares functions, constants, and any necessary global variables, making them accessible to other modules or the core AMS-DOS system.
- `.cpp` **File**: Contains the function implementations and logic for the module. This file handles the core processing, interactions with peripherals, data management, or other relevant tasks specific to the module.

3. **Example Directory Structure**:

```
/ams-dos
├── /taskmgmt
│   ├── taskmgmt.ino
│   ├── taskmgmt.h
│   └── taskmgmt.cpp
├── /memmgmt
│   ├── memmgmt.ino
│   ├── memmgmt.h
│   └── memmgmt.cpp
├── /filesys
│   ├── filesys.ino
│   ├── filesys.h
│   └── filesys.cpp
├── /periphmg
│   ├── periphmg.ino
│   ├── periphmg.h
│   └── periphmg.cpp
├── /serialio
│   ├── serialio.ino
│   ├── serialio.h
│   └── serialio.cpp
└── ... (and so on for each module)
```

4. **Modular Interaction**:

- Each module folder is self-contained, making it easy to update, test, and manage independently.
- Other modules or core files in AMS-DOS can include a module's `.h` file to access its functions and integrate its functionality, supporting seamless cross-module communication.

This repository structure ensures each module is clearly separated and manageable, making AMS-DOS a well-organized and maintainable operating system for Arduino.

5. Contribs

We invite contributors to join us in enhancing AMS-DOS. Whether you have improvements, new features, or innovative add-ons, your contributions are welcome.

AMS-DOS aims to provide a robust, adaptable operating system for Arduino, and with your help, we can extend its capabilities even further. From optimizing existing modules to introducing new functionality, every enhancement makes a difference.

Please follow our guidelines for contributing, submit your pull requests, and become part of our community dedicated to pushing the boundaries of what Arduino can achieve. Together, let's make AMS-DOS a powerful platform for all users and projects.