Storage device mapping

- SD1-8: Commodore devices (with secondary addressing 0,1 for load/save, 2-14 dynamic, 15 for commands)
- SE1-8: Serial EEPROM devices (128KB each, hardware-linked I2C addresses)
- SC1-2: SD Cards (direct mapping to SPI chip select lines)

EEPROM File System Design

Directory Structure

- Block 0 contains directory entries
- Each entry is 18 bytes:
  - +0: Control byte (status)
  - +1: 16 byte filename
  - +17: Start block pointer

Control Byte States

Value	Description
0xFF	Free (virgin) entry
0x7F	Active file
0x3F	Deleted (recoverable)
0x1F	Reallocated entry

Directory Allocation Strategy

- Scan directory entries (20 byte boundaries)
- Track first occurrence of:
  - 0xFF (free entry)
  - 0x1F (reusable entry)
  - 0x3F (deleted entry)
- Allocation priority:
  1. Try 0xFF entry with block allocation
  2. If no blocks, reclaim 0x3F chain and convert to 0x1F
  3. Reuse existing 0x1F entry

File Structure

- Files stored as sequence of counted strings
- Each string:
  - Count byte
  - String data
- End of file detected by two consecutive 0xFF bytes
- 0xFF can appear in data (check next byte)

Block Operations

- Format: Reset all blocks to 0xFF
- Directory entry reclamation:
  - Erase 18 bytes individually back to 0xFF
- Undelete:
  - Possible for 0x3F entries if chain intact
  - Convert control byte 0x3F → 0x7F

Example File Format

```
[05][Hello][05][World][01][!][FF][FF]...
 ^     ^     ^     ^     ^     ^
 count data  count data  count EOF
```

Reading Algorithm

```
While true
    count = ReadByte()
    If count == 0xFF
        next = ReadByte()
        If next == 0xFF
            // End of file
            break
        Else
            // Valid count
            Seek(-1)
        End
    End
    ReadBytes(count)
End
```

Key Benefits

- Minimal EEPROM wear
- Natural append operations
- No size tracking needed
- Simple undelete support
- Efficient space usage
- Self-describing format

SD Card File System Summary

DIRECTORY STRUCTURE Block 0: Directory entries (32 bytes each) +0: Control byte (status) 0xFF = Free (virgin) entry 0x7F = Active file 0x3F = Deleted (recoverable) 0x1F = Reallocated entry +1: Filename (29 bytes) +30: Start cluster (16-bit)

FILE STRUCTURE - Files stored as sequence of counted strings - Each string: [count byte][data bytes] - End marked by two 0xFF bytes - 0xFF in data handled by checking next byte - Clusters chained with 16-bit pointers - 0xFFFF marks end of chain

OPERATIONS - Block reads/writes (no seek needed) - 4KB buffer with block pointer navigation - Directory scans using block operations - Simple append operations - Basic undelete support - Size calculation by walking chain

BENEFITS - Minimal card wear - Natural sequential access - No file size tracking needed - Simple implementation - Efficient space usage - Self-describing format