New functions

- void init_comhand(void)
  - **Parameters**: N/A
  - **Returns**: N/A
  - **Description**: Will initiate the command handler function. Calls sys_req(READ) to get buffer user input from polling, executing code based on what the buffer reads.
- void comhand_version(void)
  - **Parameters**: N/A
  - **Returns**: N/A
  - **Description**: Will print the current project version to the user.
- void comhand_shutdown(void)
  - **Parameters**: N/A
  - **Returns**: N/A
  - **Description**: Will prompt the user for shutdown. If the user inputs 'yes', the system will initiate the shutdown procedure.
- void comhand_help(void)
  - **Parameters**: N/A
  - **Returns**: N/A
  - **Description**: Will print all currently implemented help files. Shows the user *all* commands the OS can run.
- void comhand_rtc(void)
  - **Parameters**: N/A
  - **Returns**: N/A
  - **Description**: Will print and display the current time of the real time clock to the user.
- void comhand_setTime(void)
  - **Parameters**: N/A
  - **Returns**: N/A
  - **Description**: Will prompt the user for input, asking them to enter a formatted (HH:MM:SS) text input that shows the
- getTime()
  - **Parameters**: N/A
  - **Returns**: char*
  - **Description**: will get time from ports, convert from BCD to decimal, and return as a char*

- getDate()
    - Parameters: N/A
    - Returns: char*
    - Description: gets the date from ports, converts it from bcd to int to char* and formats in it MM/DD/YY format
- int serial_poll(device dev, char* buffer, size_t len)
    - **Parameters**: device dev: the device to be read from. char* buffer: user-provided buffer. size_t len: the size of the buffer.
    - **Returns**: int the number of bytes read from the device, or a negative number on error
    - **Description**: Data is read one character at a time from the device's base register (same address as the device) using the inb() macro. When data is available, the least significant bit in the Line Status Register (device + LSR) is set
- setTime(char* newTime)
    - **Parameters**: char* newTime (containing new time to write)
    - **Returns**: N/A
    - **Description**: Takes parameter of char* and writes to the ports to set the new time
- setDate(char* newDate)
    - **Parameters**: char* newDate, the new date in MM/DD/YY format
    - **Returns**: N/A
    - **Description**: takes the date contained in newDate and writes to ports
- formatTime(int timeToFormat, char* stringPtr)
    - **Parameters**: int timeToFormat containing the time as an int read from the port. char* stringPtr as the pointer to the string to output the string to
    - **Returns**: N/A
    - **Description**: A helper function to getTime() that will take the output the int value from the ports and the pointer to the string to output, and will correctly format the time so that it shows up as ##.

- char* toupper(char* string)
    - **Parameters**: char* string (string to convert to uppercase)
    - **Returns**: New string, with each alphabetical character being uppercase.
    - **Description**: This function will turn whatever string is set as a parameter into an uppercase version of the same string. Returns a new string in memory that is fully uppercase.
- int isdigit(int c)
    - **Parameters**: int c, the character to check
    - **Returns:** 1 if c is a digit, 0 otherwise
    - **Description**: checks if c is a numeric ASCII character
- char* itoa(int i, char* dest)
    - **Parameters**: int i: the number to convert to ASCII, char* dest: location to store the resulting char* in
    - **Returns**: ASCII representation of i
    - **Description**: converts an integer to ASCII and stores the result in the memory address specified, as well as returning a reference to it.
- char* formatCore(char* format, …)
    - **Parameters**: char* format: string containing the desired string with placeholders for the rest of the arguments. Va_list: a variable length list of any elements that are to be inserted into the string.
    - **Returns**: the resulting string.
    - **Description:** takes the format string and replaces the placeholders with the specified elements
        - Use %s to insert a char*
        - Use %c to insert a char
        - Use %d or %i to insert an integer
        - Place a number between the % and the specified data type to specify the length of the string that is placed in the spot of the placeholder, if the padding size is shorter than the length of the string, the original length will be preserved. If a '.' is placed before the padding number, it will be padded with zeros as necessary.

- ➢ Int sprintf(char* dest, char* format, …)
  - ○ **Parameters**: dest: char* to store the result in. format: string containing placeholder for values, see formatCore for more details. va_list: a variable argument list of the elements to be inserted.
  - ○ **Returns**: the length of the resulting string.
  - ○ **Description**: wrapper that takes the result of formatCore and stores it in a char*. See formatCore description
- ➢ Void printf(char* format, …)
  - ○ **Parameters**: format: string containing placeholder for values, see formatCore for more details. va_list: a variable argument list of the elements to be inserted.
  - ○ **Returns**: N/A
  - ○ **Description**: wrapper that takes the result of formatCore and prints it to the console. See formatCore description for formatting details.
- ➢ Void puts(char* buf)
  - ○ **Parameters**: char* buf: a string.
  - ○ **Returns:** N/A
  - ○ **Description:** Prints the buf to the console.

<u>New data types</u>

- ➢ N/A