**Programmer’s Manual (MPX OS)**

1. **commhand():-**
   1. **Inputs:** none
   2. **Outputs:** none
   3. **Description:** Within Kernel, this function is called, which in return runs an endless loop to continuously ask the user for commands. This calls sys\_req(READ,…) which goes to polling for collecting inputs. It then uses strtok() to separate the commands into individual strings separated by spaces.
   4. **Variables:**  
      int size: size of the command buffer  
      int sizeBuff: size of the time/date buffers  
      char buffer[]: the main command buffer  
      int dateBuff[] / int timeBuff[]: the buffers to store current date and time values.  
      char \*token: pointer to the first “word” in command buffer before a space  
      const char split[]: the character that splits the command buffer
2. **get\_Time(int \*pointer):-**
   1. **inputs:** a pointer to the time buffer
   2. **outputs:** none
   3. **macros:** 0x00 defined as Sec, 0x02 defined as Min and 0x04 defined as Hr.
   4. **Description:** Grabs the hours, minutes and seconds values from their respective RTC registers and fills them in the buffer given by the pointer. Because the values in the registers are stored in BCD form, they are first converted into binary form and then stored in the buffer one digit at a time.
3. **set\_Time(int hour, int minutes, int seconds):**-
   1. **Inputs:** int hours, int minutes, int seconds
   2. **Outputs:** none
   3. **Description:** Changes the current time of the machine to the provided values. This is achieved by converting each input into BCD and then storing each input to its respective RTC register. This requires that the interrupts be disabled first before performing any changes. Once the values are stored, the interrupts are enabled again.
   4. **Variables:**   
      int s\_time: the BCD results of each input that is used to store the information to the registers.
4. **getdate(int \*ptr):**-
   1. **Inputs:** int \*pointer
   2. **Outputs:** none
   3. **Description:** Grabs the year, month and day values from the RTC registers and stores them in the buffer. Similar to get\_Time(), this function takes the values that are in BCD and converts them to regular binary, where the buffer is then filled one digit at a time.
   4. **Variables:**  
      int I / int j: the int values that would split the BCD value in half, each being converted to binary to represent the actual value to be stored.
5. **setdate(int year, int month, int day):**-
   1. **Inputs:** int year, int month, int day
   2. **Outputs:** none
   3. **Description:** Changes the current date of the machine to the provided values. Similar to set\_Time(), this converts each input into BCD form before entering the values in the registers, after disabling the interrupts. Once it is done, the interrupts are enabled again.
6. **help():**-
   1. **Inputs:** none
   2. **Outputs:** none
   3. **Description:** Once invoked in Terminal, it simply displays available commands. The messages inform the user of what commands he/she is able to perform and what syntax or parameters are needed.
7. **version():**-
   1. **Inputs:** none
   2. **Outputs:** none
   3. **Description:** Once invoked, it will print a message informing the user of the version and release date of this OS.