Creating a Custom Log File Using ScriptBasic

The DXM Controller can create custom log files using ScriptBasic. ScriptBasic supports five different file names that can be written to by the user: SbFile[1:5].dat . These files can be up to 2 GB in size, as long as they fit on the micro SD card. The data can be any string with numbers, letters, and any supported special characters.

In the ScriptBasic example below, the program uses all five files and rotates through the files to overwrite the oldest. The limits are set small to show how the program works. The log file save iteration is every 10 seconds and the file length is set to 10 lines. View the ScriptBasic output on the console to see the file and data being saved.

Two reasons to create a custom output including using a custom header (instead of register numbers) or the ability to manipulate, scale, or offset values.

Example ScriptBasic Program for Creating Custom Log Files

The first section of code defines the constants and variables used throughout the program. The second section lists the functions used for the task. The last section contains the main loop of the program. The program lines that begin with ' are comments.

```
'Save strings / numbers to a file on the SD card; this program demonstrates basic capabilities.
'Custom log files can manipulate the register data before it's written to the log file. temp = reg/20
‘Create unique headers
‘Write only under certain conditions.
'
‘*****************************************************************************
‘Constants and variables
‘*****************************************************************************
CONST LocalReg    = 199
CONST HoldingReg    = 0
CONST SaveInteration = 10
'
'File names are fixed for now and are stored in _sxi folder.
'FORMAT:  result = FILEOUT(FileIndex, Length, Flags, ContextToWrite)
' FileIndex is a target with values:
'    1 = UART
'    2 = Reserved for email
'    10 = File 1, SbFile1.dat
'    11 = File 2, SbFile2.dat
'    12= File 3, SbFile3.dat
'    13 = File 4, SbFile4.dat
'    14 = File 5, SbFile5.dat
'Length = character count of the content to write, 0 = autodetect
'Flags = 0 = Append file, 1 = Overwrite file
    ‘ContextToWrite = text string to write.
'
SbFile[ 0 ]    = 10
SbFile[ 1 ]    = 11
SbFile[ 2 ]    = 12
SbFile[ 3 ]    = 13
SbFile[ 4 ]    = 14
'
CONST AutoDetect    = 0
CONST AppendFile    = 0
CONST OverwriteFile    = 1
'
'Create a header that will define what's in the file.
CONST TextHeader     = "Vib1 Trend, Vib1 Avg, Vib1 Min, Vib1 Max, Vib2 Trend, Vib2 Avg, Vib2 Min, Vib2 Max \n"
'
'An array of registers to write out to the files
RegArray[0]     = 1
RegArray[1]     = 2
RegArray[2]     = 3
RegArray[3]     = 4
RegArray[4]     = 5
RegArray[5]     = 6
RegArray[6]     = 7
RegArray[7]     = 8
'
CONST MaxNumOfLines  = 10
'
'Variables for changing the files
NumOfLines    = 0
FileToUse    = 0
'
‘*****************************************************************************
‘Functions
‘*****************************************************************************
'Simple function for putting a custom header at the beginning of a file.
FUNCTION StartFile(FileNum, TextString)
```

```
    result = FILEOUT(FileNum, AutoDetect, OverwriteFile,TextString)
    NumOfLines = NumOfLines + 1
    PRINT " File Started: ", FileNum,"   Result: ", result, "\n  Wrote text header: ", TextString, "\n"
END FUNCTION
'
'Function to write the register contents to the file.
FUNCTION WriteFile(FileNum, RegArray)
    LOCAL SaveData
    StringOut = " T" & NOW & " ->  "
    FOR x = 0 TO UBOUND(RegArray)
        RdData     = GETREG(RegArray[x], LocalReg, HoldingReg)
        SaveData = RdData/1000
        StringOut = StringOut & " " & SaveData & ", "
    NEXT
    StringOut = StringOut & " \n"
    result        = FILEOUT(FileNum, AutoDetect, AppendFile, StringOut)
    NumOfLines = NumOfLines + 1
    PRINT " File Written: ", FileNum,"   Result: ", result, "\n  Wrote data: ", StringOut, "\n"
END FUNCTION
'
'Function to check if the max lines were written to a file, then move to the next file.
FUNCTION CheckFileSizeFUNCTION CheckFileSize
    IF NumOfLines  >= MaxNumOfLines THEN
        FileToUse       = FileToUse + 1
        NumOfLines = 0
        IF FileToUse > 4 THEN
            FileToUse = 0
        END IF
        StartFIle(SbFile[FileToUse], TextHeader)
    END IF
END FUNCTION
‘
‘*************************************************************************
‘Start of the program
‘*************************************************************************
'Start the first file (overwriting) add the header
‘
StartFIle(SbFile[FileToUse], TextHeader)
‘
WHILE(1)
    CheckFileSize
    WriteFile(SbFile[FileToUse], RegArray)
    SLEEP(SaveInteration)
WEND
END
```

Document title: Create a Custom Log File Using ScriptBasic
Part number: b_4462769
Revision: A
Original Instructions