1. **Pre- mission**

// Get input from user

Do

* (a) : Record\_interval = get\_data\_in () // get\_data\_in () maybe a pre-defined function or // defined by code writer

Return error message if data input is incorrect, prompt user to have new input

* (b): Sleep\_Interval = get\_data\_in ()

Return error message if data input is incorrect ,, prompt user to have new input

* (c) Sample\_Rate = getdatain ()

Return error message if data input is incorrect , prompt user to have new input

* Time\_to\_start = get\_data\_in ()

Return error message if data input is incorrect, prompt user to have new input

// Calculation:

Assuming Sample\_Resolution =16 bit

// File size for each single record interval

Single\_file = Sample\_Rate\*Sample\_solution\*Record\_Interval

// This step is to make sure the data storage goes smoothly, avoid any corruption in saving data

// process, may not be necessary. Good example :Single\_file = 32GB/n where n is integer n>=2.

If Single\_file> 128GB // this step can be avoided if the range of input is specified

// in get\_data\_in () function

Return error message

Go back to (a), (b), (c) Prompt user to have new input for Sample\_Rate and Recor\_Interval, Sleep\_Interval

end

If 128GB >Single\_file > 16Gb,

Return message “recommend to change Sample\_rate or Record\_duration : Yes / No”

If “Yes” go back (a),(b),(c), else continue

else

// Calculate the working time:

On\_power\_rate = constant1

Sleep\_power\_rate=constant2

//calculate working time based on power consumption

Power\_consumption\_until\_start = Sleep\_power\_rate\*(Time\_to\_start – time \_now)

Number\_of\_interval = (Total\_power\_capacity-Power\_consumtion\_until\_start) /( On\_power\_rate\*Record\_Interval + Sleep\_Power\_rate\*Sleep\_Interval)

Duration\_based\_on\_power\_consumption= (Sleep\_interval + Record\_Interval)\* number\_of\_interval;

Display on interface.

//calculate working time based on data storage capacity

Duration\_based\_on\_ storage\_capacity = (Total\_storage\_capacity /Single\_file) \*

(Record\_Interval + Sleep\_interval)

Display on interface

Working \_time =min ( duration\_based\_on\_power\_consumption,

Duration\_based\_on\_storage\_capacity)

Display on interface

Prompt user to select :

Good\_input =1 if satisfied with the schedule

Good\_input =0 if unsatisfied with schedule

While good\_input= 0;

End do while loop;

// Specify the start time, sleep time and wake-up time:

Time\_to\_start = Time\_to\_start

Start=0

Sleep=0

//Open .txt file (or any file suitable for MCU)

FileID= file\_open().. //

Fprintf (title, ….)

For i=0 to number\_of\_interval+1, i=i+1

Start= Time\_to\_start + i\* (Record\_Interval+Sleep\_Interval)

Sleep=Start+(i+1)\*Record\_Interval

Fprintf (‘setting…..’ , sample\_rate,Record\_Interval, Sleep\_Interval, Single\_file, start, sleep) // save data to file

end

close(fileID)

///////////////////////////////////////////////////////

II **In-mission**

Initialize RTC

Initialize interrupt

Flag =0;

N=0;

// Get the data from file

Load the file

Sample\_rate = read\_data\_from\_file

Record\_Interval= read\_data\_from\_file

Sleep\_Interval= read\_data\_from\_file

[ start sleep ] = read\_data\_from\_file ( FileID)

Load sample\_rate to Tascam

/////// Sleep and wake-up interrupt

For j= 0 to number\_of\_iterval +1, j=j+1

Wake\_up = Program RTC send Interrupt at start (j)

Delay by Record\_Interval

\_Go\_to\_sleep = Program RTC send interrupt at sleep (j)

Delay by Sleep\_interval

(// Not sure yet how this should work)

End

//Interrupt\_ detection function

Checking\_ interrupt

While (no interrupt) checking interrupt

If (interrupt)

check if flag=0, start system

N=N+1

Loop :

Call check\_free\_storage\_space (N, free\_space) function

If free\_space =1 Send sigal to Tascam to save data

Else,

call Mux\_switch function

N=0;

End Loop

change flag=1

If flag = 1, go to sleep

Change flag=0

end

//// Check\_free\_storage\_space function (n, free\_space)

If n < round\_down (32GB/Single\_file)-1, free\_space =yes

Else, free\_space=0;

/// Mux\_switch function

If N=round\_down(32GB/Single\_file) ,

send the signal (binary combination) to switch Mux

send signal to power down Tascam

Delay (some\_amount\_of\_time) // this is for avoiding the interference between

// interrupt and switch procedure

If flag=1

Send the signal to power on Tascam

End