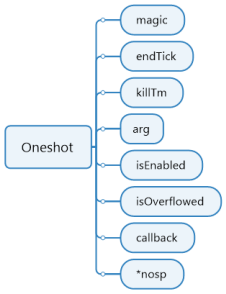
LTOS - Lightweight and Tiny Operating System

LTOS is simplified, ready to port and use, Round-robin scheduling operating system. It uses dynamic memory (malloc&free) and has simple garbage collection feature to delete unused oneshot(s) to give unnecessary memory area back to system.
There is simple struct named as oneshot. As its name, it is called just one time when it is attached to system. Oneshot structure is:



Parameter	Type	Brief
magic	uint32_t	Is used to check memory crash
endTick	tick_t	End time to call its callback function
killTm	tick_t	Periodic check time if it used or not
arg	uint32_t	Is passed as a parameter to its callback function
isEnabled	uint8_t	If it is attached or not
isOverflowed	uint8_t	If its time elapsed and ready to call its callback function
callback	os_callback_t	Callback function's pointer
*nosp	oneshot_t	Next oneshot's pointer (link list method)

There are two main files in LTOS system:

- LTOS_oneshot: To allocate, free and use (attach) oneshots.
- LTOS_tick: To get tick value in terms of tick, µs (microsecond) and ms (milisecond).

1- Porting LTOS

You only need to call **LTOS_tickIncrease** function periodically from your TIMER interrupt. That is all.

2- Using LTOS

Before use LTOS, you should set tree configurations in ***LTOS_config.h***:

LTOS_TICK_RESOL_US	It specify if LTOS uses tick resolution in terms of µs (microsecond) or ms (millisecond)
LTOS_GARBAGE_COLL_TOUT	It specify timeout for LTOS to automatically free unused oneshot(s)
LTOS_MAGIC	LTOS check memory crash by checking this value periodically and if find any crash, LTOS stops and return with **LTOS_ERR_MAGIC_CRASH** error

There are two simple steps to attach a function into chain.

- Call **LTOS_oneshotAlloc** function to allocate new oneshot to be used.
- Call **LTOS_oneshotAttach** function to attach this oneshot into LTOS Round-robin schedule. Its function will be called when timeout elapsed.

To start LTOS system, you just call **LTOS_run** function. It is endless loop. It will return if there is any memory crash while checking LTOS_MAGIC value.
Main LTOS functions are listed above:

Name	Brief	Parameter(s)		Return
LTOS_oneshotAlloc	allocates new oneshot	**None**		allocated **oneshot pointer** or **NULL**
LTOS_oneshotAttach	attach oneshot to Round-robin chain	***os**	pointer of the oneshot will be attached	**LTOS_ERR_NONE** or error code with respect to error type: **LTOS_ERR_MAGIC_CRASH**, **LTOS_ERR_INVALID_PTR**
		fp	function pointer will be called when timeout elapsed	
		arg	argument will be passed to function	
		tout	timeout to call function in terms of tick resolution	
LTOS_oneshotFree	free oneshot	***os**	Pointer of the oneshot will be freed	**LTOS_ERR_NONE** or error code with respect to error type: **LTOS_ERR_INVALID_PTR**, **LTOS_ERR_BUSY**
LTOS_run	Run LTOS forever	**None**		**LTOS_ERR_MAGIC_CRASH** if there is memory crash issue

If you want to write your delay or any other time based function by using tick value, you can also use following functions:

Name	Brief	Parameter(s)	Return
LTOS_tick2us	Get tick value in terms of µs	**None**	µs value
LTOS_ tick2ms	Get tick value in terms of ms	**None**	ms value
LTOS_getTick	Get tick value	**None**	tick value