Milestone 07-07-2020
Today I discovered and implemented a build that ported FreeRTOS to ARM Cortex M4-F based TI EK-TM4C123GXL TIVA C Series using Code Composer Studio CCS IDE

Next Step
Load the build to the board to text if it works

Approach
I started by watching a FreeRTOS port example to TIVA C using uVision (used to be Keil) https://www.youtube.com/watch?v=lj56g2fwfrQ&t=2s. The example was helpful in discovering key elements of porting FreeRTOS irrespective of IDE. I faced some issue the video didn't cover but was able to resolve after troubleshooting. I relaxed at last https://www.freertos.org/Creating-a-new-FreeRTOS-project.html would cause porting notes below to make sense.

Porting Notes

1. Get Tivaware for TI board — Has corresponding FreeRTOS files under Third party dir and also has needed startup source for ARM
2. Create an empty CCS project
3. Select the device (TM4C123GXL) of interest
4. Grab FreeRTOS sources into CCS IDE
    a. Create FreeRTOS local folder in your workspace (so you don't modify orginal download)
    b. Copy source and lic folder (these are the important FreeRTOS folders) to the local copy workspace (not actual CCS project)
    c. CCS (the compiler for the IDE) and MemMang are the only folders needed from the source>portable folder so delete all other
    d. In the CCS folder, only ARM_CM4F (the MCU arch) is needed so delete all other
    e. 6 source files are required to build a FreeRTOS project in CCS
        i. Grab 4 sources needed from FreeRTOS source folder- tasks, queue, list, timers
        ii. Grab port and portasm source from portable> CCS> port folder
        iii. Grab heap_1 source from MemMang - needed for static heap alloc
    f. Grab freeRTOSconfig.h from distro closest to your target-- I just used the config header provided in the video and modified it to
        i. adapt to my port by pound defining 3 interfaces provided by FreeRTOS and required to be initialized in the startup code init vector - Service pending interrupt handler, service handler and system tick handler
        ii. Ensured number of priority bit for interrupt level is properly defined- change prio bits to
        iii. SystemCoreClock is an extern in the config header so FreeRTOS knows the sys clock— need to define this later
5. Grab and update CCS based startup code for CM4F from your Tivaware example project to your source folder in the CCS
    a. Declare as extern the 3 interrupt handler references in startup file
    b. Update the vectors table defaults with the 3 provided references in the startup file
    c. Include <FreeRTOS.h> in the startup file so that dependencies to the provided interrupt handler can be resolved by the linker
6. Add main.c to code to CCS — create a simple freeRTOS task and run it (i used example provided in video)
    a. Include <stdint.h>, <FreeRTOS.h>, <task.h>
    b. Define SystemCoreClock to the default bus freq—as uint32_t 16M
7. Need to tell the compiler (CCS) where to find the header files provided by FreeRTOS - (ref from your local FreeRTOS copy- update compiler and linker properties to have these include paths)
    a. FreeRTOS\Source\Include
    b. FreeRTOS\Source\Portable\CCS\ARM_CM4F
8. Build CCS project and debug