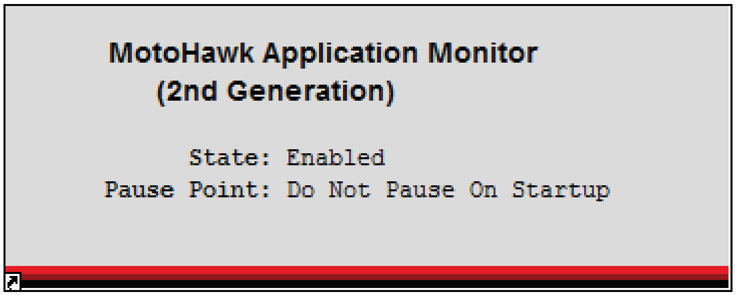
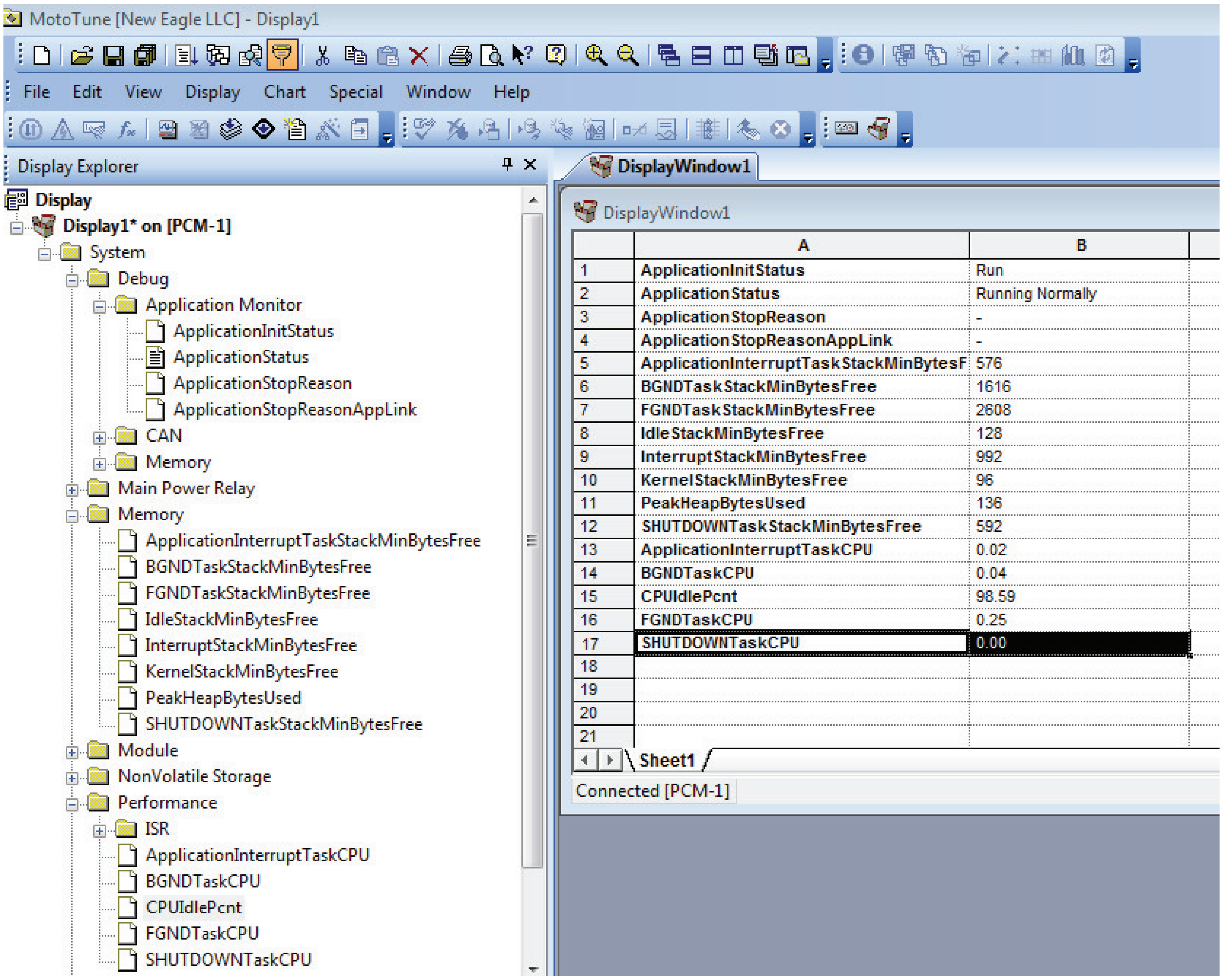
Application Monitor



MotoHawk Application Monitor
(2nd Generation)

State: Enabled
Pause Point: Do Not Pause On Startup

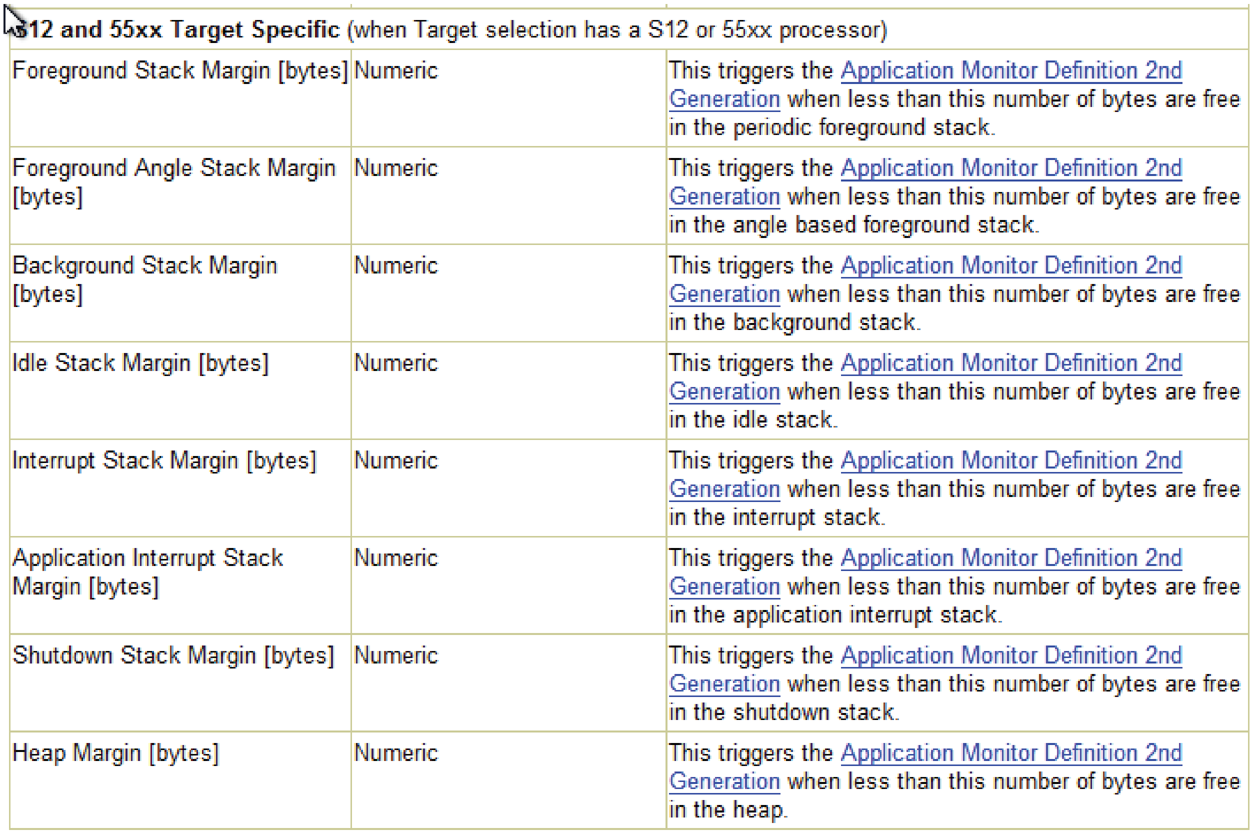
The 'MotoHawk Application Monitor' sets up a monitor within your application that checks critical system parameters as your application executes. While monitoring these parameters it will halt the application if usage exceeds certain thresholds you configure within the mask for this block. Although the application is halted when a threshold is exceeded, the MotoTune protocol handler should continue to execute meaning that you can inspect the state of the controller using MotoTune to determine what has occurred. The following MotoTune Display variables are useful in working with the App Monitor:



- Display
  - Display1* on [PCM-1]
    - System
      - Debug
        - Application Monitor
          - ApplicationInitStatus
          - ApplicationStatus
          - ApplicationStopReason
          - ApplicationStopReasonAppLink
        - CAN
        - Memory
      - Main Power Relay
      - Memory
        - ApplicationInterruptTaskStackMinBytesFree
        - BGNDTaskStackMinBytesFree
        - FGNDTaskStackMinBytesFree
        - IdleStackMinBytesFree
        - InterruptStackMinBytesFree
        - KernelStackMinBytesFree
        - PeakHeapBytesUsed
        - SHUTDOWNTaskStackMinBytesFree
      - Module
      - NonVolatile Storage
      - Performance
        - ISR
        - ApplicationInterruptTaskCPU
        - BGNDTaskCPU
        - CPUIdlePcnt
        - FGNDTaskCPU
        - SHUTDOWNTaskCPU

	A	B
1	ApplicationInitStatus	Run
2	ApplicationStatus	Running Normally
3	ApplicationStopReason	-
4	ApplicationStopReasonAppLink	-
5	ApplicationInterruptTaskStackMinBytesF	576
6	BGNDTaskStackMinBytesFree	1616
7	FGNDTaskStackMinBytesFree	2608
8	IdleStackMinBytesFree	128
9	InterruptStackMinBytesFree	992
10	KernelStackMinBytesFree	96
11	PeakHeapBytesUsed	136
12	SHUTDOWNTaskStackMinBytesFree	592
13	ApplicationInterruptTaskCPU	0.02
14	BGNDTaskCPU	0.04
15	CPUIdlePcnt	98.59
16	FGNDTaskCPU	0.25
17	SHUTDOWNTaskCPU	0.00
18		
19		
20		
21		

Configuring the Application Monitor for the S12 is done via parameters in the target definition block for most parameters. There is a separate block for the starvation timer. The parameters available are shown in the table below:



S12 and 55xx Target Specific (when Target selection has a S12 or 55xx processor)		
Foreground Stack Margin [bytes]	Numeric	This triggers the Application Monitor Definition 2nd Generation when less than this number of bytes are free in the periodic foreground stack.
Foreground Angle Stack Margin [bytes]	Numeric	This triggers the Application Monitor Definition 2nd Generation when less than this number of bytes are free in the angle based foreground stack.
Background Stack Margin [bytes]	Numeric	This triggers the Application Monitor Definition 2nd Generation when less than this number of bytes are free in the background stack.
Idle Stack Margin [bytes]	Numeric	This triggers the Application Monitor Definition 2nd Generation when less than this number of bytes are free in the idle stack.
Interrupt Stack Margin [bytes]	Numeric	This triggers the Application Monitor Definition 2nd Generation when less than this number of bytes are free in the interrupt stack.
Application Interrupt Stack Margin [bytes]	Numeric	This triggers the Application Monitor Definition 2nd Generation when less than this number of bytes are free in the application interrupt stack.
Shutdown Stack Margin [bytes]	Numeric	This triggers the Application Monitor Definition 2nd Generation when less than this number of bytes are free in the shutdown stack.
Heap Margin [bytes]	Numeric	This triggers the Application Monitor Definition 2nd Generation when less than this number of bytes are free in the heap.

The App Monitor can stop your application for the following reasons:

App Monitor Notification Stop
Application Stopped (User Command)
Starvation timer margin violation
Heap margin violation
Idle stack margin violation
Interrupt stack margin violation
ApplicationInterruptTask stack margin violation
BGNDTask stack margin violation
SHUTDOWNTask stack margin violation
FGNDTask stack margin violation

When the application monitor stops your application for one of these reasons, it is because a condition has been detected that needs to be resolved in your software. Often times, all that is required is adjusting the stack sizes to better match your application, but for applications that are approaching the resource constraints of the module, this can involve tradeoffs and optimizations.