GeneralMacros.inc

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;                                                                             ;
;                              GeneralMacros.inc                              ;
;                                CC2652  Macros                               ;
;                                                                             ;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

; This file contains generally useful macros for the CC2652 processor.  The
; macros included are:
;    MOV32 - move a 32-bit value into a register
;    MOVA  - move a 32-bit address into a register
;    STREG - store a 32-bit value in a MCU register using a base register
;
;
; Revision History
;     2/14/22  Glen George         initial revision
;    11/05/23  Glen George         updated style
;    10/28/25  Steven Lei          forked from Glen, update name for HW1


; local include files
;   none


; MOV32
;
; Description:       This macro generates code to move a 32-bit constant into
;                    a CPU register.  The macro is passed the constant and the
;                    name of the register to move it into.
;
; Operation:         The register is loaded with the low 16 bits and then the
;                    high 16-bits of the constant.
;
; Arguments:         reg   - register to be loaded with the constant value.
;                    value - 32-bit constant to store in the register.
;
; Registers Changed: None.
; Stack Depth:       0 words
;
; Author:            Glen George
; Last Modified:     Feb. 14, 2022

MOV32   .macro  reg, value

        MOVW    reg, #(value & 0×FFFF)          ;load the low 16-bits
        MOVT    reg, #((value >> 16) & 0×FFFF)  ;load the high 16-bits

```

```
48          .endm
49
50
51
52
53  ; MOVA
54  ;
55  ; Description:       This macro generates code to move a 32-bit address into
56  ;                    a CPU register.  The macro is passed the label of the
57  ;                    address and the name of the register to move it into.
58  ;
59  ; Operation:         The register is loaded with the low 16 bits and then the
60  ;                    high 16-bits of the address.
61  ;
62  ; Arguments:         reg  - register to be loaded with the address.
63  ;                    addr - 32-bit address to store in the register.
64  ;
65  ; Registers Changed: None.
66  ; Stack Depth:       0 words
67  ;
68  ; Author:            Glen George
69  ; Last Modified:     Feb. 14, 2022
70
71  MOVA    .macro  reg, addr
72
73          MOVW    reg, addr                       ;load the low 16-bits
74          MOVT    reg, addr                       ;load the high 16-bits
75
76          .endm
77
78  ; STREG
79  ;
80  ; Description:       This macro generates code to store a 32-bit constant into
81  ;                    an I/O register.  The macro is passed the constant to be
82  ;                    stored, the register containing the base address of the
83  ;                    I/O register, and the offset of the I/O register.
84  ;
85  ; Operation:         Register R0 is loaded with the low 16 bits and then the
86  ;                    high 16 bits of the value.  This is then written to the
87  ;                    specified register using the passed base register.
88  ;
89  ; Arguments:         value - 32-bit value to be stored in the I/O register.
90  ;                    base  - register containing the I/O register base address
91  ;                    off   - 16-bit offset (from the base) of the I/O register
92  ;                            to be written.
93  ;
94  ; Registers Changed: R0.
95  ; Stack Depth:       0 words
96  ;
```

```
97  ; Author:             Glen George
98  ; Last Modified:      Feb. 14, 2022
99
100 STREG   .macro  value, base, off
101
102         MOVW    R0, #(value & 0×FFFF)           ;load the low 16 bits of value
103         MOVT    R0, #((value >> 16) & 0×FFFF)   ;load high 16 bits of value
104         STR     R0, [base, #off]                ;and store the 32-bit value
105
106         .endm
```