```
; divide.asm
; Description:
;   16 and 32 bit unsigned divide routines using the SUBCU instruction
;
;   DIVIDE32 =  32 bit divide routine
;       Call routine with:  XAR0 =  Numerator
;                           XAR1 =  Denominator
;       Returns:            XAR2 =  Remainder
;                           XAR3 =  Quotient
;   DIVIDE =    Call routine with:  AR0 =   Numerator
;                           AR1 =   Denominator
;       Returns:            AR2 =   Remainder
;                           AR3 =   Quotient
;*****************************************************************************
; Begin Program:
        .global     _c_int00
WDCR    .set 0x7029             ; Watchdog Control Register

        .text
_c_int00:

    EALLOW                      ; Allow writes to GPIO regs (and others)

    SETC    objmode             ; Allows 32 bit operations

    MOV     AR0, #WDCR          ; Disable watchdog timer
    MOV     AR1, #0x68          ;
    MOV     *AR0, AR1           ;

    MOV     SP, #0x0400         ; Initialize Stack Pointer

MAINLOOP:
    ; 32 bit divide: 1,000,000/503 = 1988 (0x7C4)  r.36 (0x24)

    MOVL    XAR0, #1000000      ; Numerator32 = 1,000,000 (0xF4240)
    MOVL    XAR1, #503          ; Den32 = 503 (0x1F4)
    LC      DIVIDE32

    MOVL    XAR4, XAR2          ; save 32bit remainder (0x24=36) in XAR4
    MOVL    XAR5, XAR3          ; save 32bit quotient (0x7C4=1988) in XAR5

    ;16bit divide:   65,000 / 503 =  129 (0x 81)  r. 113 (0x71)

    MOV     AR0, #65000         ; Numerator32 = 65,000 (0xFDE8)
    MOV     AR1, #503           ; Den32 = 503 (0x1F4)
    LC      DIVIDE
                                ; Quotient (0x81=129) is in AR3
                                ; Remainder (0x71=113) is in AR2
WAIT
    B       WAIT, UNC
;*****************************************************************************
; Call with the following
;   XAR0 = Numerator
;   XAR1 = Denominator
; Return with the following
;   XAR2 = Remainder
;   XAR3 = Quotient
```

```
58 DIVIDE32:
59     PUSH    AL              ; preserve ACC
60     PUSH    AH              ;
61     PUSH    PL              ; preserve P
62     PUSH    PH              ;
63
64     MOVB    ACC,#0          ; Zero ACC
65     MOVL    P, XAR0         ; Load P register with Num32
66     RPT     #31             ; Repeat operation 32 times
67     ||SUBCUL  ACC, XAR1     ; Conditional subtract with Den32
68     MOVL    XAR2,   ACC     ; Store 32 bit remainder in XAR2
69     MOVL    XAR3, P         ; Store 32 bit quotient in XAR3
70     POP     PH              ; Restore P
71     POP     PL              ;
72     POP     AH              ; Restore ACC
73     POP     AL              ;
74
75     LRET
76 ;END OF DIVIDE
77 ;*****************************************************************************
78 ; Call with the following
79 ;   AR0 = Numerator
80 ;   AR1 = Denominator
81 ; Return with the following
82 ;   AR2 = Remainder
83 ;   AR3 = Quotient
84 DIVIDE:
85
86     PUSH    AL              ; preserve ACC
87     PUSH    AH              ;
88     MOVU    ACC, AR0        ; AL = 16 bit Numerator, AH = 0
89     RPT     #15             ; Repeat operation 16 times
90     ||SUBCU   ACC, AR1      ; Conditional subtract with Denominator
91     MOV     AR2, AH         ; Store remainder in AR2
92     MOV     AR3, AL         ; Store quotient in AR3
93
94     POP     AH              ; Restore ACC
95     POP     AL              ;
96
97     LRET
98 ;END OF DIVIDE
99 ;*****************************************************************************
100 SETUP_CPU:
101     EALLOW                  ; Allow writes to GPIO regs (and others)
102
103     SETC    objmode         ; Allows 32 bit operations
104
105     MOV     AR0, #WDCR      ; Disable watchdog timer
106     MOV     AR1, #0x68      ;
107     MOV     *AR0, AR1       ;
108
109     LRET                    ;Return
110 ;END OF SETUP_CPU
111
```