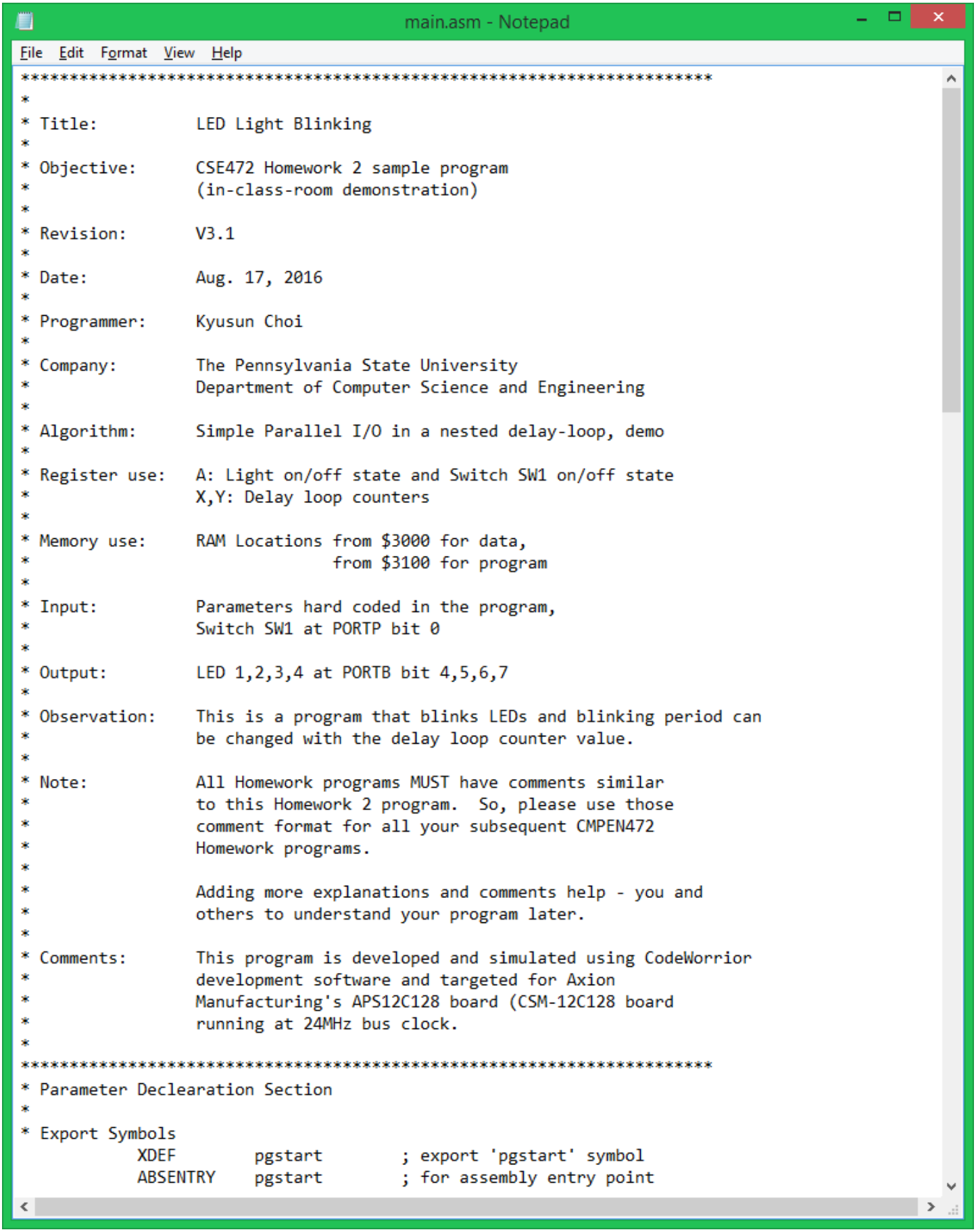
Homework 3 Aid

Understanding Homework 2/3 Program




```
*****************************************************************
*
* Title:          LED Light Blinking
*
* Objective:      CSE472 Homework 2 sample program
*                 (in-class-room demonstration)
*
* Revision:       V3.1
*
* Date:           Aug. 17, 2016
*
* Programmer:     Kyusun Choi
*
* Company:        The Pennsylvania State University
*                 Department of Computer Science and Engineering
*
* Algorithm:      Simple Parallel I/O in a nested delay-loop, demo
*
* Register use:   A: Light on/off state and Switch SW1 on/off state
*                 X,Y: Delay loop counters
*
* Memory use:     RAM Locations from $3000 for data,
*                              from $3100 for program
*
* Input:          Parameters hard coded in the program,
*                 Switch SW1 at PORTP bit 0
*
* Output:         LED 1,2,3,4 at PORTB bit 4,5,6,7
*
* Observation:    This is a program that blinks LEDs and blinking period can
*                 be changed with the delay loop counter value.
*
* Note:           All Homework programs MUST have comments similar
*                 to this Homework 2 program.  So, please use those
*                 comment format for all your subsequent CMPEN472
*                 Homework programs.
*
*                 Adding more explanations and comments help - you and
*                 others to understand your program later.
*
* Comments:       This program is developed and simulated using CodeWorrior
*                 development software and targeted for Axion
*                 Manufacturing's APS12C128 board (CSM-12C128 board
*                 running at 24MHz bus clock.
*
*****************************************************************
* Parameter Declearation Section
*
* Export Symbols
            XDEF        pgstart         ; export 'pgstart' symbol
            ABSENTRY    pgstart         ; for assembly entry point
```

main.asm - Notepad

File Edit Format View Help

```
*****************************************************************
* Parameter Declearation Section
*
* Export Symbols
            XDEF        pgstart         ; export 'pgstart' symbol
            ABSENTRY    pgstart         ; for assembly entry point
*                                       ; This is first instruction of the program
*                                       ;   up on the start of simulation

* Symbols and Macros
PORTA       EQU         $0000           ; i/o port addresses (port A not used)
DDRA        EQU         $0002

PORTB       EQU         $0001           ; PORT B is connected with LEDs
DDRB        EQU         $0003
PUCR        EQU         $000C           ; to enable pull-up mode for PORT A, B, E, K

PTP         EQU         $0258           ; PORTP data register, used for Push Switch
PTIP        EQU         $0259           ; PORTP input register <<==================
DDRP        EQU         $025A           ; PORTP data direction register
PERP        EQU         $025C           ; PORTP pull up/down enable
PPSP        EQU         $025D           ; PORTP pull up/down selection

*****************************************************************
* Data Section
*
            ORG         $3000           ; reserved RAM memory starting address
                                        ;   Memory $3000 to $30FF are for Data
Counter1    DC.W        $4fff           ; initial X register count number
Counter2    DC.W        $0020           ; initial Y register count number

StackSpace                              ; remaining memory space for stack data
                                        ; initial stack pointer position set
                                        ; to $3100 (pgstart)

*
*****************************************************************
* Program Section
*
            ORG         $3100           ;Program start address, in RAM
pgstart     LDS         #pgstart        ; initialize the stack pointer

            LDAA        #%11110000      ; set PORTB bit 7,6,5,4 as output, 3,2,1,0 as input
            STAA        DDRB            ; LED 1,2,3,4 on PORTB bit 4,5,6,7
                                        ; DIP switch 1,2,3,4 on PORTB bit 0,1,2,3.
            BSET        PUCR,%00000010  ; enable PORTB pull up/down feature, for the
                                        ; DIP switch 1,2,3,4 on the bits 0,1,2,3.

            BCLR        DDRP,%00000011  ; Push Button Switch 1 and 2 at PORTP bit 0 and 1
                                        ; set PORTP bit 0 and 1 as input
            BSET        PERP,%00000011  ; enable the pull up/down feature at PORTP bit 0 and 1
```

main.asm - Notepad

File Edit Format View Help

```
*****************************************************************
* Data Section
*
            ORG         $3000           ; reserved RAM memory starting address
                                        ;   Memory $3000 to $30FF are for Data
Counter1    DC.W        $4fff           ; initial X register count number
Counter2    DC.W        $0020           ; initial Y register count number

StackSpace



*
*****************************************************************
* Program Section
*
            ORG         $3100           ;Program start address, in RAM
pgstart     LDS         #pgstart        ; initialize the stack pointer

            LDAA        #%11110000      ; set PORTB bit 7,6,5,4 as output, 3,2,1,0 as input
            STAA        DDRB            ; LED 1,2,3,4 on PORTB bit 4,5,6,7
                                        ; DIP switch 1,2,3,4 on PORTB bit 0,1,2,3.
            BSET        PUCR,%00000010  ; enable PORTB pull up/down feature, for the
                                        ; DIP switch 1,2,3,4 on the bits 0,1,2,3.

            BCLR        DDRP,%00000011  ; Push Button Switch 1 and 2 at PORTP bit 0 and 1
                                        ; set PORTP bit 0 and 1 as input
            BSET        PERP,%00000011  ; enable the pull up/down feature at PORTP bit 0 and 1
            BCLR        PPSP,%00000011  ; select pull up feature at PORTP bit 0 and 1 for the
                                        ; Push Button Switch 1 and 2.

            LDAA        #%11110000      ; Turn off LED 1,2,3,4 at PORTB bit 4,5,6,7
            STAA        PORTB           ; Note: LED numbers and PORTB bit numbers are different

mainLoop
            BSET        PORTB,%10000000 ; Turn off LED 4 at PORTB7
            JSR         delay1sec       ; Wait for 1 second
            BCLR        PORTB,%10000000 ; Turn on LED 4 at PORTB7
            JSR         delay1sec       ; Wait for 1 second

            LDAA        PTIP            ; read push button SW1 at PORTP0
            ANDA        #%00000001      ; check the bit 0 only
            BEQ         sw1pushed
sw1notpsh   BSET        PORTB,%00010000 ; turn OFF LED1 at PORTB4
            BRA         mainLoop        ; loop forever!

sw1pushed   BCLR        PORTB,%00010000 ; turn ON LED1 at PORTB4
            BRA         mainLoop        ; loop forever!


*****************************************************************
```

Adjust this number for 1 sec. blink on your computer. $0040 will increase delay by twice long.

main.asm - Notepad

File Edit Format View Help

```
* Subroutine Section
*

;*****************************************************************
; delay1sec subroutine
;
; Please be sure to include your comments here!
;

delay1sec
            PSHY

            LDY   Counter2              ; long delay by
dly1sLoop   JSR   delay1ms              ; Y * delay1ms
            DEY
            BNE   dly1sLoop

            PULY
            RTS

;*****************************************************************
; delay1ms subroutine
;
; This subroutine cause a few msec. delay
;
; Input:  a 16bit count number in 'Counter1'
; Output: time delay, cpu cycle waisted
; Registers in use: X register, as counter
; Memory locations in use: a 16bit input number in 'Counter1'
;
; Comments: one can add more NOP instructions to lengthen
;           the delay time.

delay1ms
            PSHX

            LDX   Counter1              ; short delay
dlymsLoop   NOP                         ; X * NOP
            DEX
            BNE   dlymsLoop

            PULX
            RTS


*
* Add any more subroutines here
*


            end                         ;last line of a file
```

Delay Subroutine Timing:

$0020 ---> $0010 twice faster blinking
$4FFF ---> $0014 1000 times faster blinking
(20479) (20)

Blinking ---> Dimming

Pulse Width Modulation

50% Dimming

```
Loop   BSET     PORTB,%10000000      ; Turn OFF LED 4 at PORTB7
       JSR      delay1ms             ; Wait for 1 msecond
       BCLR     PORTB,%10000000      ; Turn ON LED 4 at PORTB7
       JSR      delay1ms             ; Wait for 1 msecond
       BRA      Loop                 ; loop forever!
```

50% Dimming

```
Loop   BSET     PORTB,%10000000      ; Turn OFF LED 4 at PORTB7
       JSR      delay1ms             ; Wait for 1 msecond
       BCLR     PORTB,%10000000      ; Turn ON LED 4 at PORTB7
       JSR      delay1ms             ; Wait for 1 msecond
       BRA      Loop                 ; loop forever!
```

Total blinking in 1 sec?

10% Dimming

```
Loop   BSET     PORTB,%10000000      ; Turn OFF LED 4 at PORTB7
       JSR      delay1ms             ; Wait for 1 msecond
       JSR      delay1ms             ; Wait for 1 msecond
       JSR      delay1ms             ; Wait for 1 msecond
       JSR      delay1ms             ; Wait for 1 msecond
       JSR      delay1ms             ; Wait for 1 msecond
       JSR      delay1ms             ; Wait for 1 msecond
       JSR      delay1ms             ; Wait for 1 msecond
       JSR      delay1ms             ; Wait for 1 msecond
       JSR      delay1ms             ; Wait for 1 msecond
       BCLR     PORTB,%10000000      ; Turn ON LED 4 at PORTB7
       JSR      delay1ms             ; Wait for 1 msecond
       BRA      Loop                 ; loop forever!
```

Total blinking in 1 sec?

6% Dimming

```
Loop   BSET     PORTB,%10000000      ; Turn OFF LED 4 at PORTB7
       JSR      delay1ms             ; Wait for 1 msecond
       JSR      delay1ms             ; Wait for 1 msecond
         •
         •                          ;  X 94
         •
       JSR      delay1ms             ; Wait for 1 msecond
       BCLR     PORTB,%10000000      ; Turn ON LED 4 at PORTB7
       JSR      delay1ms             ; Wait for 1 msecond
       JSR      delay1ms             ; Wait for 1 msecond
         •
         •                          ;  X 6
         •
       JSR      delay1ms             ; Wait for 1 msecond
       BRA      Loop                 ; loop forever!
```

Total blinking in 1 sec?

No Dimming, only blinking 10 times/sec

Revised:

6% Dimming

```
Loop   BSET     PORTB,%10000000      ; Turn OFF LED 4 at PORTB7
       JSR      delay10us             ; Wait for 10 usecond
       JSR      delay10us             ; Wait for 10 usecond
         •
         •                           ;  X 94
         •
       JSR      delay10us             ; Wait for 10 usecond
       BCLR     PORTB,%10000000      ; Turn ON LED 4 at PORTB7
       JSR      delay10us             ; Wait for 10 usecond
       JSR      delay10us             ; Wait for 10 usecond
         •
         •                           ;  X 6
         •
       JSR      delay10us             ; Wait for 10 usecond
       BRA      Loop                 ; loop forever!
```

Total blinking in 1 sec?

True Dimming, 100 times/sec

Determine how many times the instruction "SUBA" in the program below is executed if NUM=200.

```
NUM:    EQU    200
DLY:    LDAA   #NUM
LOOP:   SUBA   #$01
        BNE    LOOP
        RTS
```

_____ **times**

How long will it take to run the above subroutine DLY, running on the CSM-12C128 board with 24MHz clock? Show your work.

```
JSR     DLY
```

```
DLY:    LDAA    #1
LOOP:   SUBA    #1
        BNE     LOOP
        RTS
```

```
DLY:    LDAA    #2
LOOP:   SUBA    #1
        BNE     LOOP
        RTS
```

```
DLY:    LDAA    #3
LOOP:   SUBA    #1
        BNE     LOOP
        RTS
```

DLY:	**LDAA**	**#200**	**1 times**
LOOP:	**SUBA**	**#1**	**200 times**
	BNE	**LOOP**	**200 times**
	RTS		**1 times**

OpCode and cycle look up: Table A-2, page 395

DLY:	**LDAA**	**#200**	**1 cyc X 1 times, $86**
LOOP:	**SUBA**	**#$01**	**1 cyc X 200 times, $80**
	BNE	**LOOP**	**3 cyc X 199 times, $26**
			1 cyc X 1 times, $26
	RTS		**5 cyc X 1 times, $3D**

OpCode and cycle look up: Table A-2, page 395

DLY:	**LDAA**	**#200**	**1 cyc X 1 times, $86**
LOOP:	**SUBA**	**#$01**	**1 cyc X 200 times, $80**
	BNE	**LOOP**	**3 cyc X 199 times, $26**
			1 cyc X 1 times, $26
	RTS		**5 cyc X 1 times, $3D**

Total cycles: (1 X 1) + (1 X 200) + (3 X 199) + (1 X 1) + (5 X 1)

DLY:	**LDAA**	**#N**	**1 cyc X**	**1 times**
LOOP:	**SUBA**	**#$01**	**1 cyc X**	**N times**
	BNE	**LOOP**	**3 cyc X (N-1) times**	
			1 cyc X	**1 times**
	RTS		**5 cyc X**	**1 times**

Total cycles: (1 X 1) + (1 X N) + (3 X (N - 1)) + (1 X 1) + (5 X 1)

DLY:	LDAA	#N	1 cyc X	1 times
LOOP:	SUBA	#$01	1 cyc X	N times
	BNE	LOOP	3 cyc X (N-1) times	
			1 cyc X	1 times
	RTS		5 cyc X	1 times

Total cycles: (1 X 1) + (1 X N) + (3 X (N - 1)) + (1 X 1) + (5 X 1)

= 1 + N + (3N - 3) + 1 + 5

= 7 + 4N - 3

= 4 + 4N

```
DLY:    LDAA    #200
LOOP:   SUBA    #$01
        BNE     LOOP
        RTS
```

```
DLY:    LDAA    #200
LOOP:   SUBA    #$01
        BNE     LOOP
        RTS
```

Total Cycles = 4 + 4N = 4 + 4 * 200 = 804 cycles

How long will it take to run the following subroutine? Convert cycles to Second.

```
DLY:    LDAA    #200
LOOP:   SUBA    #$01
        BNE     LOOP
        RTS
```

Total Cycles = 804 cycles

How long will it take to run the following subroutine? Convert cycles to Second.

```
DLY:    LDAA    #200
LOOP:   SUBA    #$01
        BNE     LOOP
        RTS
```

CSM-12C128 board with 24MHz clock:

1 cycle = 1/24,000,000 Second

How long will it take to run the following subroutine? Convert cycles to Second.

```
DLY:    LDAA    #200
LOOP:   SUBA    #$01
        BNE     LOOP
        RTS
```

CSM-12C128 board with 24MHz clock:

1 cycle = 1/24,000,000 Second

804 cycles => 804/24,000,000 Second

How long will it take to run the following subroutine? Convert cycles to Second.

```
DLY:    LDAA    #200
LOOP:   SUBA    #$01
        BNE     LOOP
        RTS
```

CSM-12C128 board with 24MHz clock:

1 cycle = 1/24,000,000 Second

804 cycles => 804/24,000,000 Second = 33.5 uSec.

Find N to create 10uSec delay:

```
DLY:    LDAA    #N
LOOP:   SUBA    #$01
        BNE     LOOP
        RTS
```

Find N to create 10uSec delay:

```
DLY:    LDAA    #N
LOOP:   SUBA    #$01
        BNE     LOOP
        RTS
```

1 cycle = 1/24,000,000 Second

Find N to create 10uSec delay:

```
DLY:    LDAA    #N
LOOP:   SUBA    #$01
        BNE     LOOP
        RTS
```

1 cycle = 1/24,000,000 Second

? cycles X 1/24,000,000 Second = 10.0 uSec.

Find N to create 10uSec delay:

```
DLY:    LDAA    #N
LOOP:   SUBA    #$01
        BNE     LOOP
        RTS
```

1 cycle = 1/24,000,000 Second

240 cycles X 1/24,000,000 Second = 10.0 uSec.

Find N to create 10uSec delay:

```
DLY:    LDAA    #N
LOOP:   SUBA    #$01
        BNE     LOOP
        RTS
```

1 cycle = 1/24,000,000 Second

240 cycles X 1/24,000,000 Second = 10.0 uSec.

Total Cycles = 4 + 4N

Find N to create 10uSec delay:

```
DLY:    LDAA    #N
LOOP:   SUBA    #$01
        BNE     LOOP
        RTS
```

1 cycle = 1/24,000,000 Second

240 cycles X 1/24,000,000 Second = 10.0 uSec.

240 = 4 + 4N

Find N to create 10uSec delay:

```
DLY:    LDAA    #N
LOOP:   SUBA    #$01
        BNE     LOOP
        RTS
```

1 cycle = 1/24,000,000 Second

240 cycles X 1/24,000,000 Second = 10.0 uSec.

240 - 4 = 4N

Find N to create 10uSec delay:

```
DLY:    LDAA    #N
LOOP:   SUBA    #$01
        BNE     LOOP
        RTS
```

1 cycle = 1/24,000,000 Second

240 cycles X 1/24,000,000 Second = 10.0 uSec.

236 = 4N

Find N to create 10uSec delay:

```
DLY:    LDAA    #N
LOOP:   SUBA    #$01
        BNE     LOOP
        RTS
```

1 cycle = 1/24,000,000 Second

240 cycles X 1/24,000,000 Second = 10.0 uSec.

59 = N

Find N to create 10uSec delay:

```
DLY:    LDAA    #59
LOOP:   SUBA    #$01
        BNE     LOOP
        RTS
```

Find N to create 10uSec delay:

```
DLY:    LDAA    #$3B
LOOP:   SUBA    #$01
        BNE     LOOP
        RTS
```

Find N to create 10uSec delay:

```
DLY:    PSHA
        LDAA    #N
LOOP:   SUBA    #$01
        BNE     LOOP
        PULA
        RTS
```