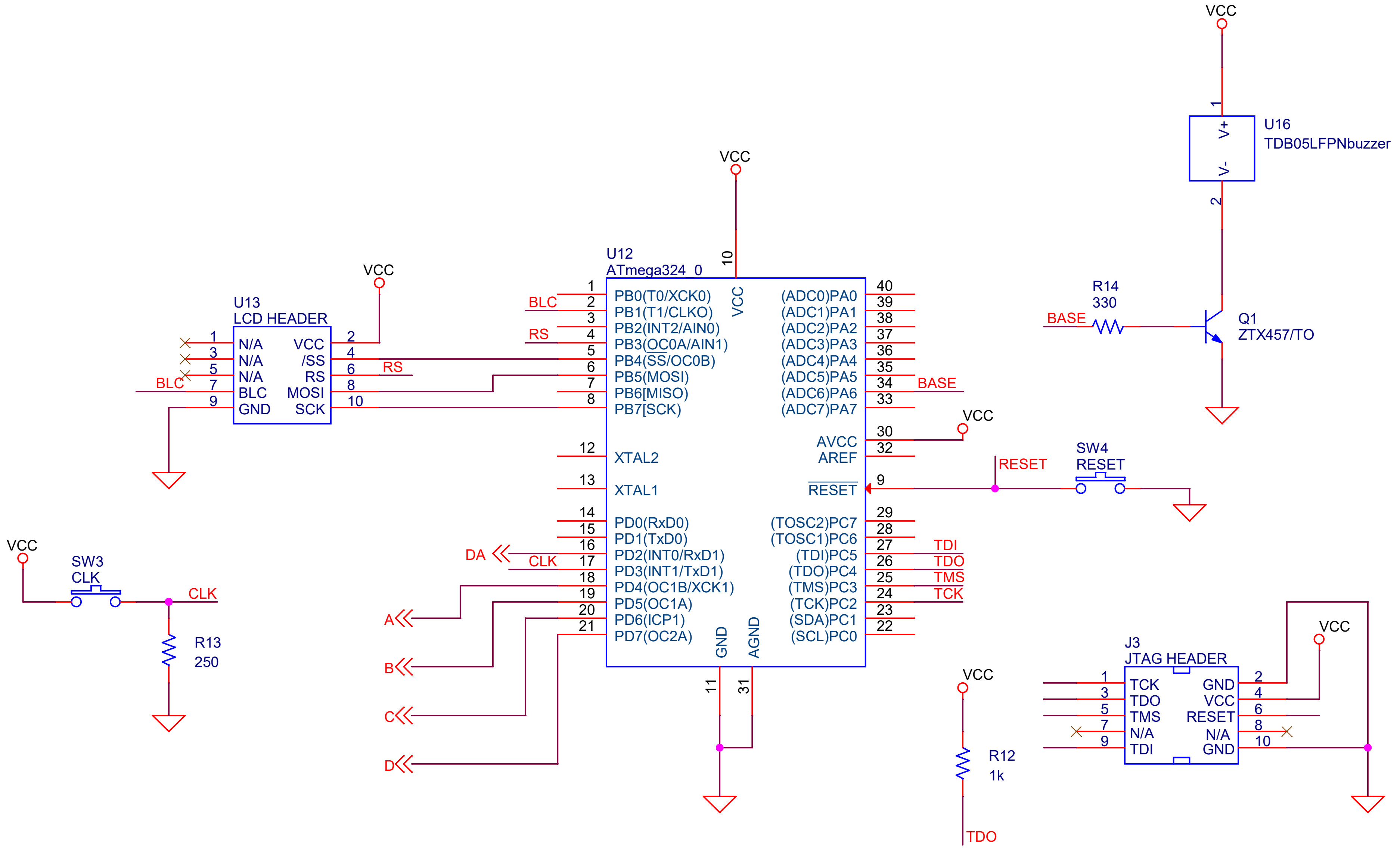
Ma & Varghese

111486364 & 111604890

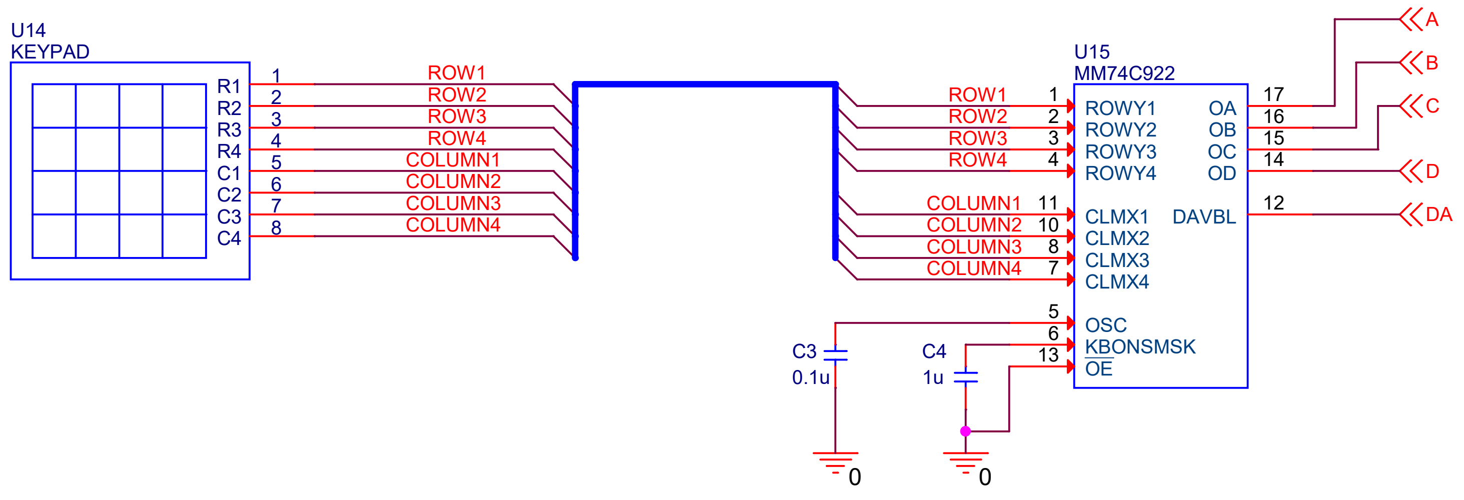
Prelab #11

L-05

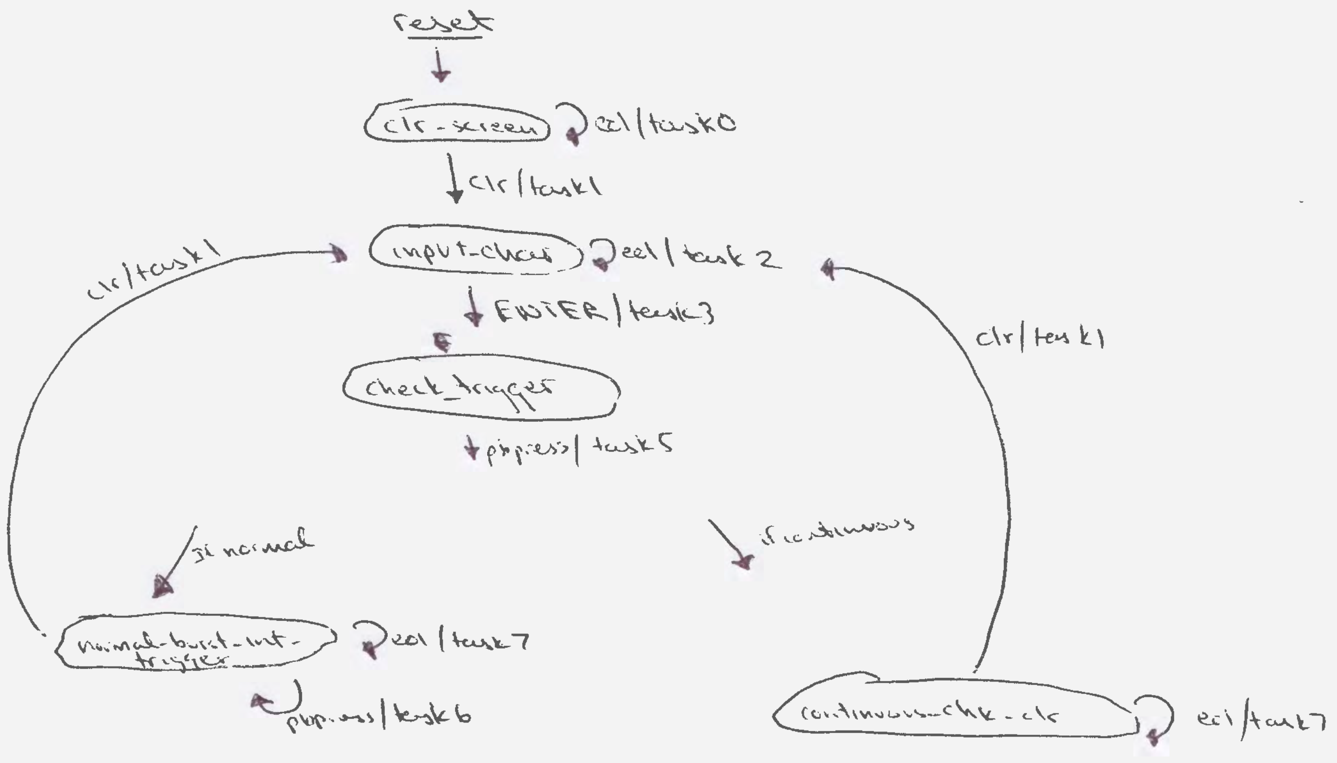
Bench #10



Title	ESE280-Prelab#10 SCHEMATIC#1 Aaron Varghese & Richard Ma
Size A	Document Number <Doc>
Rev	<RevCo
Date: Sunday, December 01, 2019	Sheet 1 of 1



Title	ESE280-Prelab#10 SCHEMATIC#2 — Aaron Varghese & Richard Ma
Size A	Document Number <Doc>
Rev	<RevCo
Date:	Sunday, December 01, 2019
Sheet	1 of 1



task0 → sounds buzzer for all other inputs pressed (not CLR)

task1 → displays n =

task 2 → inputs char into line and sounds buzzer for char that aren't digits or ENTER

task 3 → saves setting

task 4 → sounds buzzer for all inputs besides pb

task 5 → check burst count number

task 6 → reinitialize burst setting

task7 → sounds buzzer.

```

AVRASM ver. 2.2.7  C:\Users\Aaron\Desktop\College\Aaron\Junior year\Fall 2019\ESE 280\Labs\Prelab11\ ppg_IV_fsm
  \ppg_IV_fsm\ppg_IV_fsm\main.asm Thu Dec 05 18:44:32 2019

C:\Users\Aaron\Desktop\College\Aaron\Junior year\Fall 2019\ESE 280\Labs\Prelab11\ppg_IV_fsm\ppg_IV_f sm
  \ppg_IV_fsm\main.asm(15): Including file 'C:/Program Files (x86)\Atmel\Studio\7.0\Packs\atmel\ATmega _DFP
  \1.2.150\avrasm\inc\m324adef.inc'
C:\Users\Aaron\Desktop\College\Aaron\Junior year\Fall 2019\ESE 280\Labs\Prelab11\ppg_IV_fsm\ppg_IV_f sm
  \ppg_IV_fsm\main.asm(146): Including file 'C:\Users\Aaron\Desktop\College\Aaron\Junior year\Fall 201 9\ESE 280
  \Labs\Prelab11\ppg_IV_fsm\ppg_IV_fsm\ppg_IV_fsm\lcd_dog_asm_driver_m324a.inc'
C:\Users\Aaron\Desktop\College\Aaron\Junior year\Fall 2019\ESE 280\Labs\Prelab11\ppg_IV_fsm\ppg_IV_f sm
  \ppg_IV_fsm\main.asm(147): Including file 'C:\Users\Aaron\Desktop\College\Aaron\Junior year\Fall 201 9\ESE 280
  \Labs\Prelab11\ppg_IV_fsm\ppg_IV_fsm\ppg_IV_fsm\subroutines.inc'
C:\Users\Aaron\Desktop\College\Aaron\Junior year\Fall 2019\ESE 280\Labs\Prelab11\ppg_IV_fsm\ppg_IV_f sm
  \ppg_IV_fsm\main.asm(433): warning: Register r14 already defined by the .DEF directive
C:\Users\Aaron\Desktop\College\Aaron\Junior year\Fall 2019\ESE 280\Labs\Prelab11\ppg_IV_fsm\ppg_IV_f sm
  \ppg_IV_fsm\main.asm(434): warning: Register r15 already defined by the .DEF directive
C:\Users\Aaron\Desktop\College\Aaron\Junior year\Fall 2019\ESE 280\Labs\Prelab11\ppg_IV_fsm\ppg_IV_f sm
  \ppg_IV_fsm\main.asm(15): Including file 'C:/Program Files (x86)\Atmel\Studio\7.0\Packs\atmel\ATmega _DFP
  \1.2.150\avrasm\inc\m324adef.inc'
C:\Users\Aaron\Desktop\College\Aaron\Junior year\Fall 2019\ESE 280\Labs\Prelab11\ppg_IV_fsm\ppg_IV_f sm
  \ppg_IV_fsm\main.asm(146): Including file 'C:\Users\Aaron\Desktop\College\Aaron\Junior year\Fall 201 9\ESE 280
  \Labs\Prelab11\ppg_IV_fsm\ppg_IV_fsm\ppg_IV_fsm\lcd_dog_asm_driver_m324a.inc'
C:\Users\Aaron\Desktop\College\Aaron\Junior year\Fall 2019\ESE 280\Labs\Prelab11\ppg_IV_fsm\ppg_IV_f sm
  \ppg_IV_fsm\main.asm(147): Including file 'C:\Users\Aaron\Desktop\College\Aaron\Junior year\Fall 201 9\ESE 280
  \Labs\Prelab11\ppg_IV_fsm\ppg_IV_fsm\ppg_IV_fsm\subroutines.inc'


                                 ; ppg_IV_fsm.asm
                                 ;
                                 ; Created: 11/27/2019 2:28:21 PM
                                 ; Author : Aaron
                                 ; Version: 1.0
                                 ; Target: ATMEGA324A
                                 ; Description: The purpose of the program is to
                                 ; implement Lab 10 functionality(Lab 8 function
                                 ; with interrupts) except this time it will be
                                 ; implemented using the fsm chart


                                 .list


                                 .def pstatel = r24 ;low byte of present state address
                                 .def pstateh = r25 ;high byte of present state address

                                 start:
                                 .org 0x0000
000000 940c 0005                 jmp init
                                 .org INT0addr
000002 940c 00c7                 jmp ISR0
                                 .org INT1addr
000004 c0d0                      rjmp ISR1

                                 .dseg
000100                           burst_count_bcd_setting:    .byte 3 ;setting in bcd
000103                           burst_count_binary_setting:     .byte 1 ;setting in binary
000104                           normal_flag:                    .byte 1 ;1 is set
000105                           continuous_flag:                .byte 1 ; 1 is set

                                 .cseg
                                 init:
                                 ;initialize the stack pointer
000005 ef0f                      ldi r16, LOW(RAMEND)
000006 bf0d                      out SPL,r16
000007 e008                      ldi r16, HIGH(RAMEND)
000008 bf0e                      out SPH, r16

                                 ;setting up PORTB
000009 ef0f                      ldi r16, $FF     ; set portB = output.
```

```
00000a b904                      out DDRB, r16     ;
00000b 9a2c                      sbi PORTB, 4      ; set /SS of DOG LCD = 1 (Deselected)

                                 ;setting up PORTD
00000c e000                      ldi r16, $00;PortD is an input port
00000d b90a                      out DDRD, r16

00000e e003                      ldi r16, $03;initialize pull-up resistors for PD0/1
00000f b90b                      out PORTD, r16

                                 ;initialize the output PA7 for the pulses
                                 ;initialize output PA6 for the buzzer
000010 ec00                      ldi r16, $C0
000011 b901                      out DDRA, r16

                                 ;put FSM in initial state
000012 e487                      ldi pstatel, LOW(clear_screen)
000013 e091                      ldi pstateh, HIGH(clear_screen)


000014 e00f                      ldi r16, (1<<ISC00)|(1<<ISC01)|(1<<ISC10)|(1<<ISC11)
000015 9300 0069                 sts EICRA, r16

000017 e003                      ldi r16, (1<<INT0)|(1<<INT1)
000018 bb0d                      out EIMSK, r16

000019 9478                      sei

                                 main_loop:
00001a e350                      ldi r21, $30
00001b e360                      ldi r22, $30
00001c e370                      ldi r23, $30

00001d e000                      ldi r16, 0
00001e 9300 0105                 sts continuous_flag, r16
```

```
000020 9300 0104                 sts normal_flag, r16
                                 checking1:
                                 ;-------------------------------------------------
                                 ;check and see if the flag for continuous was set
000022 9100 0105                 lds r16, continuous_flag
000024 3001                      cpi r16, 1
000025 f0c9                      breq zero_burst_setting

                                 ;check to see if the flag for normal burst was set
000026 9100 0104                 lds r16, normal_flag
000028 3001                      cpi r16, 1
000029 f009                      breq normal_burst_setting
                                 ;nothing was set, so go to the beginnning
00002a f7b9                      brne checking1

                                 ;if the burst is not set at 0
                                 normal_burst_setting:
                                 ;clear the flag to show task is done
00002b e000                      ldi r16, 0
00002c 9300 0104                 sts normal_flag, r16
                                 output_normal_burst_setting:
00002e e00a                      ldi r16,10
00002f 9a17                      sbi PORTA, 7
000030 d10a                      rcall var_delay
000031 9817                      cbi PORTA,7
000032 e00a                      ldi r16,10
000033 d107                      rcall var_delay
000034 956a                      dec r22
000035 f7c1                      brne output_normal_burst_setting

                                 polling:
                                 ;check and see if setting is reinitialized
000036 9100 0104                 lds r16, normal_flag
000038 3003                      cpi r16, 3
000039 f389                      breq normal_burst_setting
```

```

                                 ;check and see if CLR is pressed
00003a 9100 0104                 lds r16, normal_flag
00003c 3002                      cpi r16, 2
00003d f2e1                      breq main_loop

00003e f7b9                      brne polling

                                 ;if the burst happens to be set at 0
                                 zero_burst_setting:
                                 ;clear the flag to show task is done
00003f e000                      ldi r16, 0
000040 9300 0105                 sts continuous_flag, r16
                                 output_zero_burst_setting:
000042 e00a                      ldi r16, 10
000043 9a17                      sbi PORTA, 7
000044 d0f6                      rcall var_delay
000045 0000                      nop
000046 0000                      nop
000047 9817                      cbi PORTA,7
000048 e00a                      ldi r16, 10
000049 d0f7                      rcall var_delay_2
                                 ;check and see if CLR is pressed
00004a 9100 0105                 lds r16, continuous_flag
00004c 3002                      cpi r16, 2
00004d f261                      breq main_loop

00004e cff3                      rjmp output_zero_burst_setting

                                 .list

                                 ;****************************************************************** *********
                                 ;*
                                 ;*Title: ISR0
                                 ;* Description: checks what button on the keypad is presseed
```

```
                                ;*
                                ;* Target:ATMEGA324A
                                ;* Number of words: 17 words
                                ;* Number of cycles: 25 cycles
                                ;*
                                ;* High registers modified:
                                ;*
                                ;* Returns: N/A
                                ;*
                                ;*also calls upon the code_to_value subroutine to decode the button  press
                                ;* returns the keycode in r18
                                ;******************************************************************* *********
                                ISR0:
0000c7 930f                     push r16
0000c8 b70f                     in r16, SREG
0000c9 930f                     push r16
0000ca b109                     in r16, PIND
0000cb 7f00                     andi r16, $F0
0000cc 9502                     swap r16
0000cd 2f20                     mov r18, r16
0000ce d019                     rcall code_to_value
0000cf 2f02                     mov r16, r18
0000d0 d097                     rcall fsm
0000d1 910f                     pop r16
0000d2 bf0f                     out SREG, r16
0000d3 910f                     pop r16
0000d4 9518                     reti

                                ;******************************************************************* *********
                                ;*
                                ;*Title: ISR1
                                ;* Description:activates when the pushbutton is pressed
                                ;*
                                ;* Target:ATMEGA324A
                                ;* Number of words: 22
```

```
                                    ;* Number of cycles: 24769
                                    ;*
                                    ;* Low registers modified: N/A
                                    ;* High registers modified: N/A

                                    ;* Returns: carry flag set or cleared
                                    ;*
                                    ;****************************************************************** *********
                                    ISR1:
0000d5 930f                         push r16
0000d6 b70f                         in r16, SREG
0000d7 930f                         push r16

                                    makedebounce:
0000d8 9b4b                         sbis PIND, 3
0000d9 cffe                         rjmp makedebounce
0000da e604                         ldi r16, 100
0000db d05f                         rcall var_delay
                                    breakdebounce:
0000dc 994b                         sbic PIND, 3
0000dd cffe                         rjmp breakdebounce
0000de e604                         ldi r16, 100
0000df d05b                         rcall var_delay

0000e0 e002                         ldi r16, (1<<INT1)
0000e1 bb0c                         out EIFR, r16
                                    ;using 16 as the pbpress
0000e2 e100                         ldi r16, $10
0000e3 d084                         rcall fsm

0000e4 910f                         pop r16
0000e5 bf0f                         out SREG, r16
0000e6 910f                         pop r16
0000e7 9518                         reti

```

```

                          ;******************************************************************* *********
                          ;*
                          ;*Title: code_to_value
                          ;* Description:function that decodes the button
                          ;*
                          ;* Target:ATMEGA324A
                          ;* Number of words: 15 words
                          ;* Number of cycles: 12 cycles
                          ;*
                          ;* High registers modified: r16, r18, ZH, ZL
                          ;* Parameters:r16
                          ;*
                          ;* Returns: N/A
                          ;*
                          ;******************************************************************* *********
                          code_to_value://table lookup
                          conversion:
0000e8 e0f1               ldi ZH, high(keyconvert*2)
0000e9 edee               ldi ZL, low(keyconvert*2)
0000ea e000               ldi r16, $00
0000eb 0fe2               add ZL, r18
0000ec 1ff0               adc ZH, r16
0000ed 9124               lpm r18, Z
0000ee 9508               ret
                          ;table of values used for all of the pushbuttons
0000ef 0201
0000f0 0f03
0000f1 0504
0000f2 0e06               keyconvert: .db $01, $02, $03,$0F, $04, $05, $06, $0E
0000f3 0807
0000f4 0d09
0000f5 000a
0000f6 0c0b                           .db $07, $08, $09, $0D, $0A, $00, $0B, $0C

```

```






                             ;******************************************************************* *********
                             ;*
                             ;*Title: update
                             ;* Description: meant to update the LCD buffer when doing the burst  prompt
                             ;*in real time
                             ;*
                             ;* Target:ATMEGA324A
                             ;* Number of words: 5
                             ;* Number of cycles: 31
                             ;*
                             ;* High registers modified: XH, XL
                             ;* Parameters:r16
                             ;*
                             ;* Returns: N/A
                             ;*
                             ;*calls prompt burst count to reinitialize that line for the n =
                             ;******************************************************************* *********

                             update:
0000f7 937d                  st X+, r23
0000f8 936d                  st X+, r22
0000f9 935d                  st X+, r21
0000fa d00e                  rcall prompt_burst_count
0000fb 9508                  ret


                             ;******************************************************************* *********
                             ;*
                             ;*Title: check_unneeded_buttons
```

```
                                 ;* Description: Checks if unneeded buttons were pressed on the keyp ad
                                 ;*
                                 ;* Target:ATMEGA324A
                                 ;* Number of words: 10
                                 ;* Number of cycles:14
                                 ;*
                                 ;* High registers modified: r21
                                 ;*
                                 ;* Returns: N/A
                                 ;*
                                 ;******************************************************************* *********
                                 check_unneeded_buttons:
0000fc 305b                      cpi r21, $0B
0000fd f039                      breq equal

0000fe 305d                      cpi r21, $0D
0000ff f029                      breq equal

000100 305e                      cpi r21, $0E
000101 f019                      breq equal

000102 305f                      cpi r21, $0F
000103 f009                      breq equal

000104 c000                      rjmp notequal

                                 equal:

                                 notequal:
000105 9508                      ret

                                 ;******************************************************************* *********
                                 ;*
                                 ;*Title: convert_hex
                                 ;* Description: simple program used to convert hex
```

```
                               ;*
                               ;* Target:ATMEGA324A
                               ;* Number of words: 3
                               ;* Number of cycles: 6
                               ;*
                               ;* High registers modified: r17, r21
                               ;* Parameters:r16
                               ;*
                               ;* Returns: r19
                               ;*
                               ;***************************************************************** *********
                               convert_hex:
000106 e310                    ldi r17, $30
000107 0f51                    add r21, r17
000108 9508                    ret

                               ;***************************************************************** *********
                               ;*
                               ;*Title: prompt_burst_count
                               ;* Description: Creates n<space>=<space> on the LCD
                               ;* also used in order to set the X pointer pointing to the start of  the
                               ;*numbers to be displayed on the LCD
                               ;*
                               ;* Target:ATMEGA324A
                               ;* Number of words: 11
                               ;* Number of cycles: 18
                               ;*
                               ;*
                               ;* High registers modified: r16, XH, XL
                               ;*
                               ;* Parameters:r16
                               ;*
                               ;* Returns: N/A
                               ;*
                               ;***************************************************************** *********
```

```
                                  prompt_burst_count:
000109 e0b1                       ldi XH, high(dsp_buff_1)
00010a e0a6                       ldi XL, low(dsp_buff_1)

00010b e60e                       ldi r16, $6E ;displays n
00010c 930d                       st X+, r16

00010d e200                       ldi r16, $20;displays <space>
00010e 930d                       st X+, r16

00010f e30d                       ldi r16, $3D ;displays =
000110 930d                       st X+, r16

000111 e200                       ldi r16, $20 ;displays <space>
000112 930d                       st X+, r16
000113 df8e                       rcall update_lcd_dog
                                  ;at the end, should display n<space>=<space>
                                  ;on the LCD
000114 9508                       ret



                                  ;****************************************************************** *********
                                  ;*
                                  ;* "BCD2bin16" - BCD to 16-Bit Binary Conversion
                                  ;*
                                  ;* This subroutine converts a 5-digit packed BCD number represented  by
                                  ;* 3 bytes (fBCD2:fBCD1:fBCD0) to a 16-bit number (tbinH:tbinL).
                                  ;* MSD of the 5-digit number must be placed in the lowermost nibble  of fBCD2.
                                  ;*
                                  ;* Let "abcde" denote the 5-digit number. The conversion is done by
                                  ;* computing the formula: 10(10(10(10a+b)+c)+d)+e.
                                  ;* The subroutine "mul10a"/"mul10b" does the multiply-and-add opera tion
                                  ;* which is repeated four times during the computation.
                                  ;*
```

```
                                        ;* Number of words :30
                                        ;* Number of cycles    :108
                                        ;* Low registers used  :4 (copyL,copyH,mp10L/tbinL,mp10H/tbinH)
                                        ;* High registers used  :4 (fBCD0,fBCD1,fBCD2,adder)
                                        ;*
                                        ;***************************************************************** *********

                                        ;***** "mul10a"/"mul10b" Subroutine Register Variables

                                        .def    copyL   =r12        ;temporary register
                                        .def    copyH   =r13        ;temporary register
                                        .def    mp10L   =r14        ;Low byte of number to be multiplied by 10
                                        .def    mp10H   =r15        ;High byte of number to be multiplied by 10
                                        .def    adder   =r19        ;value to add after multiplication

                                        ;***** Code

                                        mul10a:    ;***** multiplies "mp10H:mp10L" with 10 and adds "adder" hi gh nibble
000115 9532                                swap    adder
                                        mul10b:    ;***** multiplies "mp10H:mp10L" with 10 and adds "adder" lo w nibble
000116 2cce                                mov copyL,mp10L ;make copy
000117 2cdf                                mov copyH,mp10H
000118 0cee                                lsl mp10L       ;multiply original by 2
000119 1cff                                rol mp10H
00011a 0ccc                                lsl copyL       ;multiply copy by 2
00011b 1cdd                                rol copyH
00011c 0ccc                                lsl copyL       ;multiply copy by 2 (4)
00011d 1cdd                                rol copyH
00011e 0ccc                                lsl copyL       ;multiply copy by 2 (8)
00011f 1cdd                                rol copyH
000120 0cec                                add mp10L,copyL ;add copy to original
000121 1cfd                                adc mp10H,copyH
000122 703f                                andi    adder,0x0f   ;mask away upper nibble of adder
000123 0ee3                                add mp10L,adder ;add lower nibble of adder
000124 f408                                brcc    m10_1       ;if carry not cleared
```

```
000125 94f3                          inc mp10H       ;   inc high byte
000126 9508                       m10_1: ret

                                  ;***** Main Routine Register Variables

                                  .def    tbinL   =r14        ;Low byte of binary result (same as mp10L)
                                  .def    tbinH   =r15        ;High byte of binary result (same as mp10H)
                                  .def    fBCD0   =r16        ;BCD value digits 1 and 0
                                  .def    fBCD1   =r17        ;BCD value digits 2 and 3
                                  .def    fBCD2   =r18        ;BCD value digit 5

                                  ;***** Code

                                  BCD2bin16:
000127 702f                          andi    fBCD2,0x0f  ;mask away upper nibble of fBCD2
000128 24ff                          clr mp10H
000129 2ee2                          mov mp10L,fBCD2 ;mp10H:mp10L = a
00012a 2f31                          mov adder,fBCD1
00012b dfe9                          rcall   mul10a      ;mp10H:mp10L = 10a+b
00012c 2f31                          mov adder,fBCD1
00012d dfe8                          rcall   mul10b      ;mp10H:mp10L = 10(10a+b)+c
00012e 2f30                          mov adder,fBCD0
00012f dfe5                          rcall   mul10a      ;mp10H:mp10L = 10(10(10a+b)+c)+d
000130 2f30                          mov adder,fBCD0
000131 dfe4                          rcall   mul10b      ;mp10H:mp10L = 10(10(10(10a+b)+c)+d)+e
000132 9508                          ret

                                  ;************************
                                  ;NAME:      clr_dsp_buffs
                                  ;FUNCTION:  Initializes dsp_buffers 1, 2, and 3 with blanks (0x20)
                                  ;ASSUMES:   Three CONTIGUOUS 16-byte dram based buffers named
                                  ;           dsp_buff_1, dsp_buff_2, dsp_buff_3.
                                  ;RETURNS:   nothing.
                                  ;MODIFIES:  r25,r26, Z-ptr
                                  ;CALLS:     none
```

```
                               ;CALLED BY: main application and diagnostics
                               ;*********************************************************************** **
                               clr_dsp_buffs:
000133 e390                        ldi R25, 48                ; load total length of both buffer.
000134 e2a0                        ldi R26, ' '               ; load blank/space into R26.
000135 e0f1                        ldi ZH, high (dsp_buff_1) ; Load ZH and ZL as a pointer to 1st
000136 e0e6                        ldi ZL, low (dsp_buff_1)  ; byte of buffer for line 1.

                                   ;set DDRAM address to 1st position of first line.
                               store_bytes:
000137 93a1                        st  Z+, R26        ; store ' ' into 1st/next buffer byte and
                                                      ; auto inc ptr to next location.
000138 959a                        dec  R25           ;
000139 f7e9                        brne store_bytes   ; cont until r25=0, all bytes written.
00013a 9508                        ret

                                   ;********************************************************************* ******

                               ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;; ;;;;;;;;;
                               ;DELAY FUNCTIONS
                               ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;; ;;;;;;;;;


                               ;*********************************************************************** *********
                               ;*
                               ;*Title: var_delay
                               ;* Description: Creates a delay for the ATMEGA324(has to run @ 1Mhz  CLK)
                               ;*
                               ;* Target:ATMEGA324A
                               ;* Number of words:6
                               ;* Number of cycles: 25248 cycles (when n = 32 and m = 255)
                               ;* n = inner loop variable and m = outer loop variable
                               ;* 3(nm+m+1)= total number of cycles
                               ;* When n=0, then number of cycles : 3(nm+m+n+3)
                               ;*
```

```
                                 ;* High registers modified: r16, r17
                                 ;* Parameters:r16
                                 ;*
                                 ;* Returns: N/A
                                 ;*
                                 ;*Delay provided should be around n*0.1ms
                                 ;*As stated before, n is the number inputted into r16
                                 ;***************************************************************** *********
                                 var_delay: ;delay for ATmega324 @ 1MHz = r16 * 0.1 ms
                                 outer_loop:
    00013b e210                  ldi r17, 32
                                 inner_loop:
    00013c 951a                  dec r17
    00013d f7f1                  brne inner_loop
    00013e 950a                  dec r16
    00013f f7d9                  brne outer_loop
    000140 9508                  ret


                                 ;*
                                 ;*Title: var_delay_2
                                 ;* Description: Creates a delay for the ATMEGA324(has to run @ 1Mhz  CLK)
                                 ;* Same purpose as var_delay, except that r17 is defined as 31
                                 ;* Target:ATMEGA324A
                                 ;* Number of words:6
                                 ;* Number of cycles:
                                 ;* n = inner loop variable and m = outer loop variable
                                 ;* 3(nm+m+1)= total number of cycles
                                 ;* When n=0, then number of cycles : 3(nm+m+n+3)
                                 ;*
                                 ;* High registers modified: r16, r17
                                 ;* Parameters:r16
                                 ;*
                                 ;* Returns: N/A
                                 ;*
```

```
                                ;*Delay provided should be around n*0.1ms
                                ;*As stated before, n is the number inputted into r16
                                ;******************************************************************* *********
                                var_delay_2:
                                outer:
000141 e11f                     ldi r17, 31
                                inner:
000142 951a                     dec r17
000143 f7f1                     brne inner
000144 950a                     dec r16
000145 f7d9                     brne outer
000146 9508                     ret

                                ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;; ;;;;;;;;;
                                ;FSM
                                ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;; ;;;;;;;;;
                                ;******************************************************************* *********
                                ;*
                                ;* "fsm" - Simplified Table Driven Finite State Machine
                                ;*
                                ;* Description:
                                ;* This table driven FSM can handle 255 or fewer input symbols.
                                ;*
                                ;* Author:       Aaron
                                ;* Version:      1.0
                                ;* Last updated: 12/1/2019
                                ;* Target:              ATmega324a
                                ;* Total number of cycles depends on the task being called, which i s why it
                                ;* is excluded here
                                ;* Total number of words: 54 words
                                ;* Low regs modified:   r16, r18, r20, r21, r31, and r31
                                ;* High registers used:
                                ;*
                                ;* Parameters:          present state in r25:r24 prior to call
                                ;*                      input symbol in r16 prior to call
```

```
                                   ;*
                                   ;* Notes: Calls upon the stub of tasks given in "taskn"
                                   ;*
                                   ;****************************************************************** *********

                                   ;input symbols for example finite state machine
                                   .equ i0 = $00   ;input symbols equated to numerical values ;
                                   .equ i1 = $01
                                   .equ i2 = $02
                                   .equ i3 = $03
                                   .equ i4 = $04
                                   .equ i5 = $05
                                   .equ i6 = $06
                                   .equ i7 = $07
                                   .equ i8 = $08
                                   .equ i9 = $09
                                   .equ UP = $0F
                                   .equ DOWN = $0E
                                   .equ key2nd = $0D
                                   .equ CLEAR = $0A
                                   .equ HELP = $0B
                                   .equ ENTERED = $0C
                                   .equ pbpress = $10

                                   .equ eol = $FF  ;end of list (subtable) do not change

                                   ;state table
                                   ;each row consists of input symbol, next state address, task
                                   ;subroutine address

                                   state_table:

                                   clear_screen:
000147 000a
000148 014d
```

```
000149 0186                         .dw CLEAR,      input_char,     task1
00014a 00ff
00014b 0147
00014c 017c                          .dw eol,    clear_screen,       task0

                                  input_char:
00014d 000c
00014e 0153
00014f 01ab                         .dw ENTERED,        check_trigger_pressed,      task3
000150 00ff
000151 014d
000152 018f                          .dw eol,    input_char,     task2
                                  check_trigger_pressed:
000153 0010
000154 0000
000155 01c7                         .dw pbpress, 0 , task5
000156 00ff
000157 0153
000158 01c2                         .dw eol,    check_trigger_pressed,   task4
                                  normalburst_check_buttons:
000159 000a
00015a 014d
00015b 0186                         .dw CLEAR,      input_char,     task1
00015c 0010
00015d 0159
00015e 01d8                         .dw pbpress, normalburst_check_buttons ,task6
00015f 00ff
000160 0159
000161 01e0                         .dw eol, normalburst_check_buttons, task7
                                  continuous_check_clr:
000162 000a
000163 014d
000164 0186                         .dw CLEAR,      input_char,     task1
000165 00ff
000166 0162
```

```
000167 01e0                        .dw eol,        continuous_check_clr,    task7


                                fsm:
                                ;load Z with a byte pointer to the subtable corresponding to the
                                ;present state
000168 2fe8                         mov ZL, pstatel ;load Z pointer with pstate address * 2
000169 0fee                         add ZL, ZL ;since Z will be used as a byte pointer with the lpm  instr.
00016a 2ff9                         mov ZH, pstateh
00016b 1fff                         adc ZH, ZH

                                ;search subtable rows for input symbol match
                                search:
00016c 9124                         lpm r18, Z ;get symbol from state table
00016d 1720                         cp r18, r16 ;compare table entry with input symbol
00016e f021                         breq match

                                ;check input symbol against eol
                                check_eol:
00016f 3f2f                         cpi r18, eol ;compare low byte of table entry with eol
000170 f011                         breq match

                                nomatch:
000171 9636                         adiw ZL, $06 ;adjust Z to point to next row of state table
000172 cff9                         rjmp search ;continue searching

                                ;a match on input value to row input value has been found
                                ;the next word in this row is the next state address
                                ;the word following that is the task subroutine's address
                                match:
                                   ;make preseent state equal to next state value in row
                                   ;this accomplishes the stat transition
000173 9632                         adiw ZL, $02 ;point to low byte of state address
000174 9185                         lpm pstatel, Z+; ;copy next state addr. from table to preseent  stat
000175 9195                         lpm pstateh, Z+

                                          ;execute the subroutine that accomplihes the task associated
                                          ;with the transition
000176 9135                                lpm r19, Z+ ;get subroutine address from state table
000177 9144                                lpm r20, Z ;and put it in Z pointer
000178 2fe3                                mov ZL, r19
000179 2ff4                                mov ZH, r20
00017a 9509                                icall ;Z pointer is now used as a word pointer
00017b 9508                                ret



                                       ;****************************************************************** *********
                                       ;*
                                       ;* "taskn" - Stub subroutines for testing
                                       ;*
                                       ;* Description: These are the tasks called by the FSM state table
                                       ;* Each task has a specific way of being defined, depending on wher e in the
                                       ;* state diagram it is being used
                                       ;*
                                       ;* Author: Aaron Varghese
                                       ;* Version: 1.0
                                       ;* Last updated: 12/1/2019
                                       ;* Target:ATMEGA324A
                                       ;*
                                       ;*
                                       ;*
                                       ;* Notes: The registers that is used for each task is different, an d the
                                       ;*number of words and cycles for each task is different, which is w hy the
                                       ;*information is excluded
                                       ;*
                                       ;****************************************************************** *********


                                       ;subroutine stubs for tasks to be implemented
```

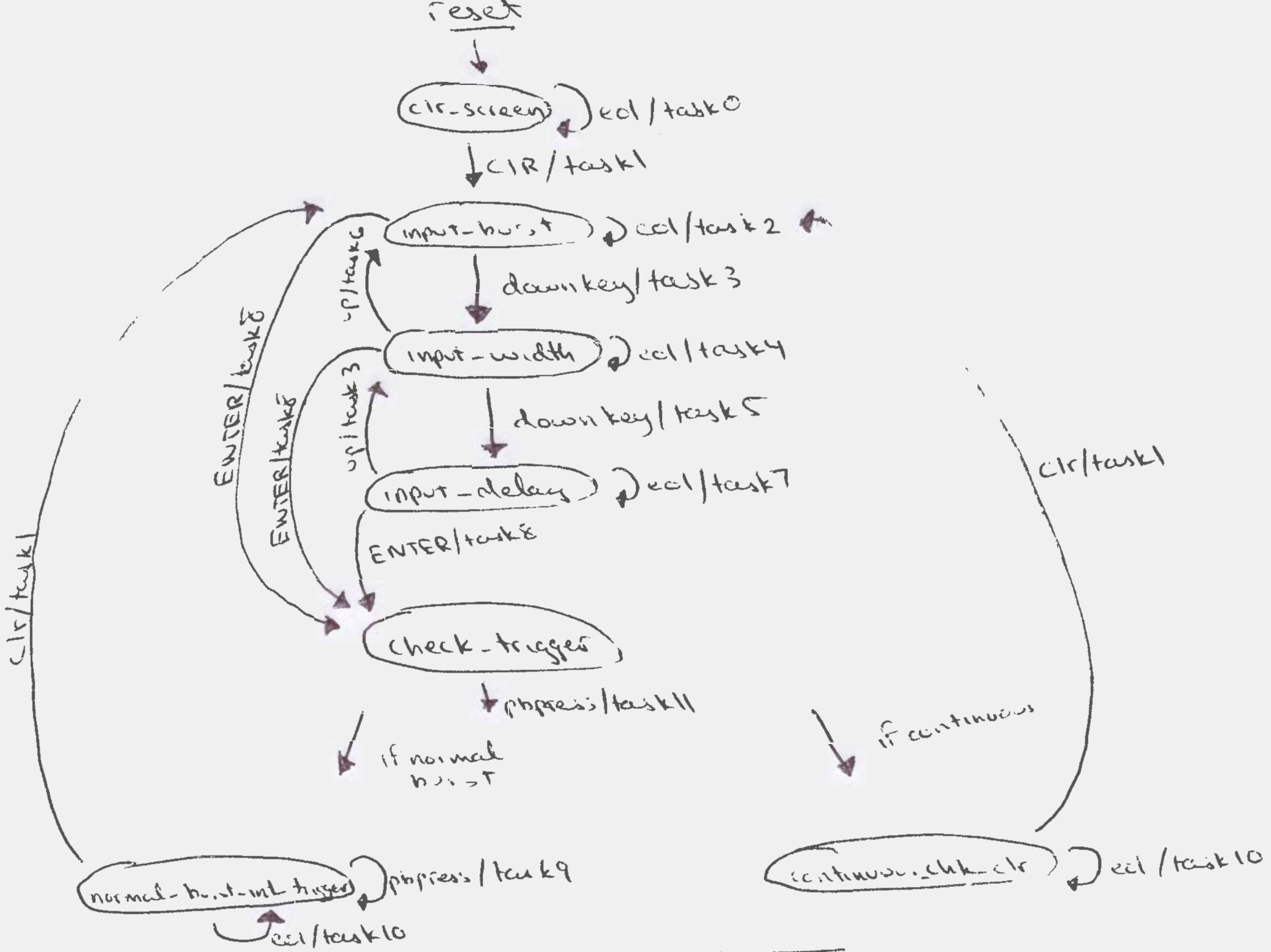
```

                                   ;clears the screen, and sound buzzer for inputs that aren't clear
                                   ;add stuff to sound buzzer
                                   task0:
                                   ;sounds the buzzer for all other inputs pressed besides CLR
 00017c ef0f                       ldi r16, 255
 00017d 9a16                       sbi PORTA, 6
 00017e dfbc                       rcall var_delay
 00017f 9816                       cbi PORTA, 6
                                   ;clears the screen
 000180 dee0                       rcall init_spi_lcd;initialize lcd screen
 000181 dfb1                       rcall clr_dsp_buffs;displays a blank screen
 000182 df1f                       rcall update_lcd_dog;updates the screen


                                   ;put FSM in initial state
 000183 e487                       ldi pstatel, LOW(clear_screen)
 000184 e091                       ldi pstateh, HIGH(clear_screen)

 000185 9508                       ret

                                   ;displays the prompt, when clear is pressed
                                   task1:
 000186 e002                       ldi r16, 2
 000187 9300 0104                  sts normal_flag, r16

 000189 e002                       ldi r16, 2
 00018a 9300 0105                  sts continuous_flag, r16

                                   ;prompt for the burst count
 00018c df7c                       rcall prompt_burst_count
 00018d df14                       rcall update_lcd_dog
 00018e 9508                       ret

                                   ;input numbers into the screen, and sound buzzer for
                                   ;characters that aren't digits
                                   task2:
                                   ;getting the value of the burst count
                                   ;--------------------------------------------------
                                   ;meant to run in an infinite loop getting values
                                   ;until enter is pressed
                                   ;r23-->MSB
                                   ;r21-->LSB
                                   inner_loop1:
00018f 2f20                        mov r18, r16

000190 302c                        cpi r18, ENTERED
000191 f099                        breq end

000192 302f                        cpi r18,UP
000193 f091                        breq endwithbuzzer

000194 302e                        cpi r18,DOWN
000195 f081                        breq endwithbuzzer

000196 302d                        cpi r18,key2nd
000197 f071                        breq endwithbuzzer

000198 302b                        cpi r18,HELP
000199 f061                        breq endwithbuzzer

00019a 302a                        cpi r18,CLEAR
00019b f051                        breq endwithbuzzer

00019c 3120                        cpi r18, pbpress
00019d f041                        breq endwithbuzzer

00019e 2f76                        mov r23, r22
00019f 2f65                        mov r22, r21

```

```
0001a0 2f52                     mov r21, r18

0001a1 df64                     rcall convert_hex


0001a2 df66                     rcall prompt_burst_count
0001a3 df53                     rcall update
0001a4 defd                     rcall update_lcd_dog
                                ;when the enter key is pressed
                                end:
0001a5 9508                     ret

                                ;turn on buzzer for all other inputs that
                                ;can't be used
                                endwithbuzzer:
0001a6 ef0f                     ldi r16, 255
0001a7 9a16                     sbi PORTA, 6
0001a8 df92                     rcall var_delay
0001a9 9816                     cbi PORTA, 6
0001aa 9508                     ret


                                ;saves the setting (both binary and bcd setting)
                                task3:
                                ;added in case enter is pressed without inputting
                                ;numbers (000 is default)
0001ab e0d1                     ldi YH, high(burst_count_bcd_setting)
0001ac e0c0                     ldi YL, low(burst_count_bcd_setting)
0001ad df5b                     rcall prompt_burst_count
0001ae df48                     rcall update
0001af def2                     rcall update_lcd_dog
                                ;if enter is pressed
                                ;storing burst_count_bcd_setting
0001b0 e0f1                     ldi ZH, high(burst_count_bcd_setting)
0001b1 e0e0                     ldi ZL, low(burst_count_bcd_setting)
```

```
                                ;convert back into unpacked hex
0001b2 5370                     subi r23, $30
0001b3 5360                     subi r22, $30
0001b4 5350                     subi r21, $30
                                ;storing it into the setting
0001b5 9371                     st Z+, r23
0001b6 9361                     st Z+, r22
0001b7 9351                     st Z+, r21

                                ;prepping for the BCD2BIN function -->43210
0001b8 9562                     swap r22
                                ;will give BCD digits 1 and 0
0001b9 2f05                     mov r16, r21
0001ba 0f06                     add r16, r22
                                ;will give BCD digits 3 and 2
0001bb 2f17                     mov r17, r23
                                ;set BCD digit 4 and 5 as 0
0001bc e020                     ldi r18, $00

0001bd df69                     rcall BCD2bin16

0001be e0d1                     ldi YH, high(burst_count_binary_setting)
0001bf e0c3                     ldi YL, low(burst_count_binary_setting)
                                ;has the value of binary setting stored in Y
0001c0 82e8                     st Y, r14
0001c1 9508                     ret

                                ;sounds the buzzer for inputs other than pb press
                                task4:
0001c2 ef0f                     ldi r16, 255
0001c3 9a16                     sbi PORTA, 6
0001c4 df76                     rcall var_delay
0001c5 9816                     cbi PORTA, 6
0001c6 9508                     ret

```

```
                                 ;check burst count number
                                 task5:
0001c7 90e0 0103                 lds r14, burst_count_binary_setting
0001c9 2d6e                      mov r22, r14
0001ca 3060                      cpi r22, 0
0001cb f031                      breq continuous_flag_set

                                 normal_flag_set:
0001cc e001                      ldi r16,1
0001cd 9300 0104                 sts normal_flag, r16
0001cf e589                      ldi pstatel, low(normalburst_check_buttons)
0001d0 e091                      ldi pstateh, high(normalburst_check_buttons)
0001d1 9508                      ret

                                 continuous_flag_set:
0001d2 e001                      ldi r16,1
0001d3 9300 0105                 sts continuous_flag, r16
0001d5 e682                      ldi pstatel, low(continuous_check_clr)
0001d6 e091                      ldi pstateh, high(continuous_check_clr)
0001d7 9508                      ret

                                 ;reinitialize burst setting
                                 task6:
0001d8 e0d1                      ldi YH, high(burst_count_binary_setting)
0001d9 e0c3                      ldi YL, low(burst_count_binary_setting)
                                 ;has the value of binary setting stored in Y
0001da 80e8                      ld r14, Y
0001db 2d6e                      mov r22, r14

                                 ;3 means that you redo the burst setting
0001dc e003                      ldi r16, 3
0001dd 9300 0104                 sts normal_flag, r16
0001df 9508                      ret

                                 ;meant to sound the buzzer for all other
                                 ;inputs besides pbpress and CLR
                                 task7:
0001e0 ef0f                      ldi r16, 255
0001e1 9a16                      sbi PORTA, 6
0001e2 df58                      rcall var_delay
0001e3 9816                      cbi PORTA, 6


RESOURCE USE INFORMATION
------------------------

Notice:
The register and instruction counts are symbol table hit counts,
and hence implicitly used resources are not counted, eg, the
'lpm' instruction without operands implicitly uses r0 and z,
none of which are counted.

x,y,z are separate entities in the symbol table and are
counted separately from r26..r31 here.

.dseg memory usage only counts static data declared with .byte

"ATmega324A" register use summary:
x  :   7 y  :   2 z  :  13 r0 :   0 r1 :   0 r2 :   0 r3 :   0 r4 :   0
r5 :   0 r6 :   0 r7 :   0 r8 :   0 r9 :   0 r10:   0 r11:   0 r12:   5
r13:   5 r14:  10 r15:   5 r16: 126 r17:  10 r18:  19 r19:   9 r20:  10
r21:  12 r22:  14 r23:   8 r24:  10 r25:   8 r26:   3 r27:   1 r28:   3
r29:   3 r30:  13 r31:  11
Registers used: 23 out of 35 (65.7%)

"ATmega324A" instruction use summary:
.lds  :   0 .sts  :   0 adc   :   3 add   :   6 adiw  :   2 and   :   0
andi  :   3 asr   :   0 bclr  :   0 bld   :   0 brbc  :   0 brbs  :   0
brcc  :   1 brcs  :   0 break :   0 breq  :  19 brge  :   0 brhc  :   0
brhs  :   0 brid  :   0 brie  :   0 brlo  :   0 brlt  :   0 brmi  :   0
```

```
brne  :  14 brpl  :   0 brsh  :   0 brtc  :   0 brts  :   0 brvc  :   0
brvs  :   0 bset  :   0 bst   :   0 call  :   0 cbi   :   9 cbr   :   0
clc   :   0 clh   :   0 cli   :   0 cln   :   0 clr   :   1 cls   :   0
clt   :   0 clv   :   0 clz   :   0 com   :   0 cp    :   1 cpc   :   0
cpi   :  18 cpse  :   0 dec   :  12 eor   :   0 fmul  :   0 fmuls :   0
fmulsu:   0 icall :   1 ijmp  :   0 in    :  12 inc   :   1 jmp   :   2
ld    :   4 ldd   :   0 ldi   :  87 lds   :   6 lpm   :   9 lsl   :   4
lsr   :   0 mov   :  21 movw  :   0 mul   :   0 muls  :   0 mulsu :   0
neg   :   0 nop   :   4 or    :   0 ori   :   0 out   :  13 pop   :  10
push  :  10 rcall :  67 ret   :  29 reti  :   2 rjmp  :   8 rol   :   4
ror   :   0 sbc   :   0 sbci  :   0 sbi   :  13 sbic  :   1 sbis  :   1
sbiw  :   0 sbr   :   0 sbrc  :   0 sbrs  :   2 sec   :   0 seh   :   0
sei   :   1 sen   :   0 ser   :   0 ses   :   0 set   :   0 sev   :   0
sez   :   0 sleep :   0 spm   :   0 st    :  12 std   :   0 sts   :  10
sub   :   0 subi  :   3 swap  :   3 tst   :   0 wdr   :   0
Instructions used: 40 out of 113 (35.4%)

"ATmega324A" memory use summary [bytes]:
Segment   Begin    End      Code   Data   Used    Size   Use%
---------------------------------------------------------------
[.cseg] 0x000000 0x0003ca    888     82    970   32768   3.0%
[.dseg] 0x000100 0x000136      0     54     54    2048   2.6%
[.eseg] 0x000000 0x000000      0      0      0    1024   0.0%

Assembly complete, 0 errors, 2 warnings

```



task0 --> sounds ~~all~~ buzzer for inputs press (not CLR), clrs screen

task1 —> displays all prompts, makes burst line active

task 2 —> inputs digits into display buffer (burst) sounds buzzer for other digits besides ENTER

task 3 —> width line active (clr active line for all other lines)

task 4 —> inputs digits into width line, sounds buzzer for digits (not ENTER)

task 5 —> delay line active

task 6 —> burst line active

task 7 —> input digits into delay line, sounds buzzer for digits (not ENTER)

task 8 —> saves all settings

task 9 —> reinitialize burst settings

task 10 —> sounds buzzer.

task 11 —> check if burst condition is 0 or normal.

```

AVRASM ver. 2.2.7  C:\Users\Aaron\Desktop\College\Aaron\Junior year\Fall 2019\ESE 280\Labs\Prelab11
  \ppg_IV_fsm_extra\ppg_IV_fsm_extra\ppg_IV_fsm_extra\main.asm Thu Dec 05 18:49:39 2019

C:\Users\Aaron\Desktop\College\Aaron\Junior year\Fall 2019\ESE 280\Labs\Prelab11\ppg_IV_fsm_extra
  \ppg_IV_fsm_extra\ppg_IV_fsm_extra\main.asm(13): Including file 'C:/Program Files (x86)\Atmel\Studio \7.0
  \Packs\atmel\ATmega_DFP\1.2.150\avrasm\inc\m324adef.inc'
C:\Users\Aaron\Desktop\College\Aaron\Junior year\Fall 2019\ESE 280\Labs\Prelab11\ppg_IV_fsm_extra
  \ppg_IV_fsm_extra\ppg_IV_fsm_extra\main.asm(167): Including file 'C:\Users\Aaron\Desktop\College\Aar on\Junior
   year\Fall 2019\ESE 280\Labs\Prelab11\ppg_IV_fsm_extra\ppg_IV_fsm_extra\ppg_IV_fsm_extra
  \lcd_dog_asm_driver_m324a.inc'
C:\Users\Aaron\Desktop\College\Aaron\Junior year\Fall 2019\ESE 280\Labs\Prelab11\ppg_IV_fsm_extra
  \ppg_IV_fsm_extra\ppg_IV_fsm_extra\main.asm(168): Including file 'C:\Users\Aaron\Desktop\College\Aar on\Junior
   year\Fall 2019\ESE 280\Labs\Prelab11\ppg_IV_fsm_extra\ppg_IV_fsm_extra\ppg_IV_fsm_extra\subroutines .inc'
C:\Users\Aaron\Desktop\College\Aaron\Junior year\Fall 2019\ESE 280\Labs\Prelab11\ppg_IV_fsm_extra
  \ppg_IV_fsm_extra\ppg_IV_fsm_extra\main.asm(502): warning: Register r14 already defined by the .DEF  directive
C:\Users\Aaron\Desktop\College\Aaron\Junior year\Fall 2019\ESE 280\Labs\Prelab11\ppg_IV_fsm_extra
  \ppg_IV_fsm_extra\ppg_IV_fsm_extra\main.asm(503): warning: Register r15 already defined by the .DEF  directive
C:\Users\Aaron\Desktop\College\Aaron\Junior year\Fall 2019\ESE 280\Labs\Prelab11\ppg_IV_fsm_extra
  \ppg_IV_fsm_extra\ppg_IV_fsm_extra\main.asm(13): Including file 'C:/Program Files (x86)\Atmel\Studio \7.0
  \Packs\atmel\ATmega_DFP\1.2.150\avrasm\inc\m324adef.inc'
C:\Users\Aaron\Desktop\College\Aaron\Junior year\Fall 2019\ESE 280\Labs\Prelab11\ppg_IV_fsm_extra
  \ppg_IV_fsm_extra\ppg_IV_fsm_extra\main.asm(167): Including file 'C:\Users\Aaron\Desktop\College\Aar on\Junior
   year\Fall 2019\ESE 280\Labs\Prelab11\ppg_IV_fsm_extra\ppg_IV_fsm_extra\ppg_IV_fsm_extra
  \lcd_dog_asm_driver_m324a.inc'
C:\Users\Aaron\Desktop\College\Aaron\Junior year\Fall 2019\ESE 280\Labs\Prelab11\ppg_IV_fsm_extra
  \ppg_IV_fsm_extra\ppg_IV_fsm_extra\main.asm(168): Including file 'C:\Users\Aaron\Desktop\College\Aar on\Junior
   year\Fall 2019\ESE 280\Labs\Prelab11\ppg_IV_fsm_extra\ppg_IV_fsm_extra\ppg_IV_fsm_extra\subroutines .inc'


                                  ; ppg_IV_fsm_extra.asm
                                  ;
                                  ; Created: 12/1/2019 11:33:18 PM
                                  ; Author : Aaron
                                  ;Target:ATMEGA324A
```

```
                                 ; Created: 11/3/2019 11:16:51 PM
                                 ;Description: Using the ATMEGA324A, we will be creating a system
                                 ;that would allow the user to implement functionality from lab9
                                 ;using interrupts and the fsm table

                                 .list
                                    `

                                 .def pstatel = r24 ;low byte of present state address
                                 .def pstateh = r25 ;high byte of present state address

                                 .dseg
000100                           burst_count_bcd_setting:   .byte 3 ;setting in bcd
000103                           burst_count_binary_setting:    .byte 1 ;setting in binary

000104                           pulse_width_bcd_setting:   .byte 3 ;setting in bcd
000107                           pulse_width_binary_setting:    .byte 1 ;setting in binary

000108                           delay_time_bcd_setting:    .byte 3 ;setting in bcd
00010b                           delay_time_binary_setting: .byte 1 ;setting in binary

00010c                           continuous_flag: .byte 1 ;flag for continuous burst
00010d                           normal_flag: .byte 1; flag for the normal burst

                                 .cseg
                                 start:
                                 .org 0x0000
000000 c005                      rjmp init
                                 .org int0addr
000002 940c 00d1                 jmp ISR0
                                 .org int1addr
000004 940c 00df                 jmp ISR1


                                 init:
                                  ;initialize the stack pointer
000006 ef0f                       ldi r16, LOW(RAMEND)
000007 bf0d                       out SPL,r16
000008 e008                       ldi r16, HIGH(RAMEND)
000009 bf0e                       out SPH, r16

                                  ;setting up PORTB
00000a ef0f                       ldi r16, $FF     ; set portB = output.
00000b b904                       out DDRB, r16     ;
00000c 9a2c                       sbi PORTB, 4      ; set /SS of DOG LCD = 1 (Deselected)

                                  ;setting up PORTD
00000d e000                       ldi r16, $00;PortD is an input port
00000e b90a                       out DDRD, r16

00000f e003                       ldi r16, $03;initialize pull-up resistors for PD0-1
000010 b90b                       out PORTD, r16

                                  ;setting up PORTA
000011 ec00                       ldi r16, $C0
000012 b901                       out DDRA, r16

000013 e30f                       ldi r16, $3F;initialize pull-up resistors for PD0-1
000014 b90b                       out PORTD, r16



                                  ;initializing the interrupts
                                  ;configures positive edge triggering
000015 e00f                       ldi r16, (1<<ISC00)|(1<<ISC01)|(1<<ISC10)|(1<<ISC11)
000016 9300 0069                  sts EICRA, r16
                                  ;configures each of the interrupts
000018 e003                       ldi r16, (1<<INT0)|(1<<INT1)
000019 bb0d                       out EIMSK, r16

```

```
00001a d050                      rcall init_spi_lcd;initialize lcd screen

00001b d12b                      rcall clr_dsp_buffs;displays a blank screen

00001c d08f                      rcall update_lcd_dog;updates the screen

                                 ;put FSM in initial state
00001d e98a                      ldi pstatel,low(clr_screen)
00001e e091                      ldi pstateh, high(clr_screen)
00001f 9478                      sei

                                 main_loop:
                                 ;polling to see if any of the flags are set
000020 9100 010d                 lds r16, normal_flag
000022 3001                      cpi r16, 1
000023 f029                      breq normal_burst_setting
000024 9100 010c                 lds r16, continuous_flag
000026 3001                      cpi r16, 1
000027 f0d9                      breq zero_burst_setting

000028 cff7                      rjmp main_loop

                                 ;if the burst is not set at 0
                                 ;gives a 1 of 998 cycles positive and
                                 ;gives a 0 of around 99-1000 cycles
                                 normal_burst_setting:
000029 e000                      ldi r16, 0
00002a 9300 010d                 sts normal_flag, r16
                                 ;time delay
00002c e0d1                      ldi YH, high(delay_time_binary_setting)
00002d e0cb                      ldi YL, low(delay_time_binary_setting)
00002e 80f8                      ld r15, Y
                                 ;pulse width
00002f e0d1                      ldi YH, high(pulse_width_binary_setting)
000030 e0c7                      ldi YL, low(pulse_width_binary_setting)
```

```
000031 80d8                      ld r13, Y
                                 output_normal_burst_setting:
000032 2d0d                      mov r16, r13
000033 9a17                      sbi PORTA, 7
000034 d126                      rcall var_delay
000035 9817                      cbi PORTA,7
000036 2d0f                      mov r16, r15
000037 d123                      rcall var_delay
000038 956a                      dec r22
000039 f7c1                      brne output_normal_burst_setting

                                 polling:
                                 ;check and see if setting is reinitialized
00003a 9100 010d                 lds r16, normal_flag
00003c 3003                      cpi r16, 3
00003d f359                      breq normal_burst_setting

                                 ;check and see if CLR is pressed
00003e 9100 010d                 lds r16, normal_flag
000040 3002                      cpi r16, 2
000041 f2f1                      breq main_loop

000042 f7b9                      brne polling

                                 ;if the burst happens to be set at 0
                                 zero_burst_setting:
000043 e000                      ldi r16, 0
000044 9300 010c                 sts continuous_flag, r16
                                 ;time delay
000046 e0d1                      ldi YH, high(delay_time_binary_setting)
000047 e0cb                      ldi YL, low(delay_time_binary_setting)
000048 80f8                      ld r15, Y
                                 ;pulse width
000049 e0d1                      ldi YH, high(pulse_width_binary_setting)
00004a e0c7                      ldi YL, low(pulse_width_binary_setting)
```

```
00004b 80d8                         ld r13, Y
                                    ;gives a 1 of 1000 cycles and a 0 of 1007 cycles
                                    output_zero_burst_setting:
00004c 2d0d                         mov r16, r13
00004d 9a17                         sbi PORTA, 7
00004e d10c                         rcall var_delay
00004f 0000                         nop
000050 0000                         nop
000051 9817                         cbi PORTA,7
000052 2d0f                         mov r16, r15
000053 d107                         rcall var_delay
000054 9100 010c                    lds r16, continuous_flag
000056 3002                         cpi r16, 2
000057 f241                         breq main_loop

000058 cff3                         rjmp output_zero_burst_setting

                                    .list

                                    ;****************************************************************** *********
                                    ;*
                                    ;*Title: ISR0
                                    ;* Description: checks what button on the keypad is presseed
                                    ;*
                                    ;* Target:ATMEGA324A
                                    ;* Number of words: 17 words
                                    ;* Number of cycles: 25 cycles
                                    ;*
                                    ;* High registers modified:
                                    ;*
                                    ;* Returns: N/A
                                    ;*
                                    ;*also calls upon the code_to_value subroutine to decode the button  press
                                    ;* returns the keycode in r18
                                    ;****************************************************************** *********
```

```
                                  ISR0:
0000d1 930f                       push r16
0000d2 b70f                       in r16, SREG
0000d3 930f                       push r16
0000d4 b109                       in r16, PIND
0000d5 7f00                       andi r16, $F0
0000d6 9502                       swap r16
0000d7 2f20                       mov r18, r16
0000d8 d02b                       rcall code_to_value
0000d9 2f02                       mov r16, r18
0000da d0f8                       rcall fsm
0000db 910f                       pop r16
0000dc bf0f                       out SREG, r16
0000dd 910f                       pop r16
0000de 9518                       reti

                                  ;****************************************************************** *********
                                  ;*
                                  ;*Title: ISR1
                                  ;* Description: Checks to see if the pushbutton is activated
                                  ;*
                                  ;* Target:ATMEGA324A
                                  ;* Number of words: 22
                                  ;* Number of cycles: 24769
                                  ;*
                                  ;* Low registers modified: N/A
                                  ;* High registers modified: N/A

                                  ;* Returns: carry flag set or cleared
                                  ;*
                                  ;****************************************************************** *********
                                  ISR1:
0000df 930f                       push r16
0000e0 b70f                       in r16, SREG
0000e1 930f                       push r16
```

```

                                  makedebounce:
0000e2 9b4b                       sbis PIND, 3
0000e3 cffe                       rjmp makedebounce
0000e4 e604                       ldi r16, 100
0000e5 d075                       rcall var_delay
                                  breakdebounce:
0000e6 994b                       sbic PIND, 3
0000e7 cffe                       rjmp breakdebounce
0000e8 e604                       ldi r16, 100
0000e9 d071                       rcall var_delay

0000ea e002                       ldi r16, (1<<INT1)
0000eb bb0c                       out EIFR, r16
                                  ;using 16 as the pbpress
0000ec e100                       ldi r16, $10
0000ed d0e5                       rcall fsm

0000ee 910f                       pop r16
0000ef bf0f                       out SREG, r16
0000f0 910f                       pop r16
0000f1 9518                       reti
                                  ;******************************************************************* *********
                                  ;*
                                  ;*Title: decode_button
                                  ;* Description: checks what button on the keypad is preseed
                                  ;*
                                  ;* Target:ATMEGA324A
                                  ;* Number of words: 12 words
                                  ;* Number of cycles: 31 cycles
                                  ;*
                                  ;* High registers modified: r16, r18, ZH, ZL
                                  ;*
                                  ;*also calls upon the code_to_value subroutine to decode the button  press
                                  ;*returns the decoded keycode from the table into r18
```

```
                                    ;******************************************************************* *********
                                    decode_button:
                                    ;check to see if a button is pressed
0000f2 9488                         clc ;indicates that button is not pressed
0000f3 9b36                         sbis PINC, 6
0000f4 9508                         ret
0000f5 b109                         in r16, PIND
0000f6 7f00                         andi r16, $F0
0000f7 9502                         swap r16
0000f8 2f20                         mov r18, r16
0000f9 d00a                         rcall code_to_value
0000fa 9408                         sec ;indicates that a key is pressed
                                    ;clears the D-FF and sets it again for polling
0000fb 9847                         cbi PORTC, 7
0000fc 9a47                         sbi PORTC, 7
0000fd 9508                         ret

                                    ;******************************************************************* *********
                                    ;*
                                    ;*Title: select_line
                                    ;* Description: meant to show which line is currently active and
                                    ;*writing inputs to the LCD
                                    ;*
                                    ;* Target:ATMEGA324A
                                    ;* Number of words: 3 words
                                    ;* Number of cycles: 7 cycles
                                    ;*
                                    ;* High registers modified: r18, YH, YL
                                    ;*
                                    ;* Returns: N/A
                                    ;*
                                    ;******************************************************************* *********
                                    select_line:
0000fe e223                         ldi r18, '#'
0000ff 872b                         std Y+11, r18
```

```
000100 9508                     ret

                                ;****************************************************************** *********
                                ;*
                                ;*Title: deselect_line
                                ;* Description: meant to show which line is currently inactive
                                ;* Target:ATMEGA324A
                                ;* Number of words: 3 words
                                ;* Number of cycles: 7 cycles
                                ;*
                                ;* High registers modified: r18, YH, YL
                                ;*
                                ;* Returns: N/A
                                ;*
                                ;****************************************************************** *********
                                deselect_line:
000101 e220                     ldi r18, ' '
000102 872b                     std Y+11, r18
000103 9508                     ret


                                ;****************************************************************** *********
                                ;*
                                ;*Title: code_to_value
                                ;* Description:function that decodes the button
                                ;*
                                ;* Target:ATMEGA324A
                                ;* Number of words: 15 words
                                ;* Number of cycles: 12 cycles
                                ;*
                                ;* High registers modified: r16, r18, ZH, ZL
                                ;* Parameters:r16
                                ;*
                                ;* Returns: N/A
                                ;*
```

```
                                 ;****************************************************************** *********
                                 code_to_value://table lookup
                                 conversion:
000104 e0f2                      ldi ZH, high(keyconvert*2)
000105 e1e6                      ldi ZL, low(keyconvert*2)
000106 e000                      ldi r16, $00
000107 0fe2                      add ZL, r18
000108 1ff0                      adc ZH, r16
000109 9124                      lpm r18, Z
00010a 9508                      ret
                                 ;table of values used for all of the pushbuttons
00010b 0201
00010c 0f03
00010d 0504
00010e 0e06                      keyconvert: .db $01, $02, $03,$0F, $04, $05, $06, $0E
00010f 0807
000110 0d09
000111 000a
000112 0c0b                                  .db $07, $08, $09, $0D, $0A, $00, $0B, $0C




                                 ;****************************************************************** *********
                                 ;*
                                 ;*Title: keep_values
                                 ;* Description: meant to take in the values in a line after switchi ng in
                                 ;* between lines
                                 ;*used to prevent lines from being written with other line's values  in them
                                 ;* Target:ATMEGA324A
                                 ;* Number of words: 4 words
                                 ;* Number of cycles: 10 cycles
                                 ;*
                                 ;* High registers modified: r21,r22,r23
                                 ;*
                                 ;* Returns: Previous values of the a current line
                                 ;*
                                 ;* meant to be used after initializing the prompt for a specific li ne
                                 ;* using either prompt_burst_count or prompt_pulse_width_count
                                 ;****************************************************************** *********
                                 keep_values:
000113 9179                      ld r23, Y+
000114 9169                      ld r22, Y+
000115 9159                      ld r21, Y+
000116 9508                      ret



                                 ;****************************************************************** *********
                                 ;*
                                 ;*Title: check_unneeded_buttons
                                 ;* Description: Checks if unneeded buttons were pressed on the keyp ad
                                 ;* Unneeded buttons are now the 2nd and HELP buttons
                                 ;*
                                 ;* Target:ATMEGA324A
                                 ;* Number of words: 10 words
                                 ;* Number of cycles:14 cycles
                                 ;*
                                 ;* High registers modified: r18
                                 ;*
                                 ;* Returns: N/A
                                 ;*
                                 ;****************************************************************** *********
                                 check_unneeded_buttons:
000117 302b                      cpi r18, $0B
000118 f019                      breq equal

000119 302d                      cpi r18, $0D
00011a f009                      breq equal

```

```
00011b c000                     rjmp notequal

                                equal:

                                notequal:
00011c 9508                     ret

                                ;******************************************************************* *********
                                ;*
                                ;*Title: convert_hex
                                ;* Description: simple program used to convert hex
                                ;*
                                ;* Target:ATMEGA324A
                                ;* Number of words: 3 words
                                ;* Number of cycles: 6 cycles
                                ;*
                                ;* High registers modified: r17, r21
                                ;* Parameters:r16
                                ;*
                                ;* Result is in: r21
                                ;*
                                ;******************************************************************* *********
                                convert_hex:
00011d e310                     ldi r17, $30
00011e 0f51                     add r21, r17
00011f 9508                     ret




                                ;******************************************************************* *********
                                ;*
                                ;*Title: check_internal_trigger
                                ;* Description:checking if Q at PA3 is set
                                ;*
```

```
                                   ;* Target:ATMEGA324A
                                   ;* Number of words: 9 words
                                   ;* Number of cycles: 24769 cycles
                                   ;*
                                   ;* High registers modified: r16
                                   ;* Returns: carry flag set or cleared
                                   ;*
                                   ;***************************************************************** *********
                                   check_internal_trigger:
000120 9488                        clc ;carry clear means that the pushbutton is not pressed
000121 9b03                        sbis PINA,3
000122 9508                        ret;pushbutton is not pressed

000123 9408                        sec; pushbutton has been pressed
000124 9812                        cbi PORTA, 2
000125 ef0a                        ldi r16, 250
000126 d034                        rcall var_delay
000127 9a12                        sbi PORTA, 2
000128 9508                        ret


                                   ;***************************************************************** *********
                                   ;*
                                   ;* "BCD2bin16" - BCD to 16-Bit Binary Conversion
                                   ;*
                                   ;* This subroutine converts a 5-digit packed BCD number represented  by
                                   ;* 3 bytes (fBCD2:fBCD1:fBCD0) to a 16-bit number (tbinH:tbinL).
                                   ;* MSD of the 5-digit number must be placed in the lowermost nibble  of fBCD2.
                                   ;*
                                   ;* Let "abcde" denote the 5-digit number. The conversion is done by
                                   ;* computing the formula: 10(10(10(10a+b)+c)+d)+e.
                                   ;* The subroutine "mul10a"/"mul10b" does the multiply-and-add opera tion
                                   ;* which is repeated four times during the computation.
                                   ;*
                                   ;* Number of words  :30
                                   ;* Number of cycles    :108
                                   ;* Low registers used  :4 (copyL,copyH,mp10L/tbinL,mp10H/tbinH)
                                   ;* High registers used  :4 (fBCD0,fBCD1,fBCD2,adder)
                                   ;*
                                   ;***************************************************************** *********

                                   ;***** "mul10a"/"mul10b" Subroutine Register Variables

                                   .def    copyL   =r12        ;temporary register
                                   .def    copyH   =r13        ;temporary register
                                   .def    mp10L   =r14        ;Low byte of number to be multiplied by 10
                                   .def    mp10H   =r15        ;High byte of number to be multiplied by 10
                                   .def    adder   =r19        ;value to add after multiplication

                                   ;***** Code

                                   mul10a:    ;***** multiplies "mp10H:mp10L" with 10 and adds "adder" hi gh
                                   ;nibble
  000129 9532                         swap    adder
                                   mul10b:    ;***** multiplies "mp10H:mp10L" with 10 and adds "adder" lo w
                                   ;nibble
  00012a 2cce                         mov copyL,mp10L ;make copy
  00012b 2cdf                         mov copyH,mp10H
  00012c 0cee                         lsl mp10L       ;multiply original by 2
  00012d 1cff                         rol mp10H
  00012e 0ccc                         lsl copyL       ;multiply copy by 2
  00012f 1cdd                         rol copyH
  000130 0ccc                         lsl copyL       ;multiply copy by 2 (4)
  000131 1cdd                         rol copyH
  000132 0ccc                         lsl copyL       ;multiply copy by 2 (8)
  000133 1cdd                         rol copyH
  000134 0cec                         add mp10L,copyL ;add copy to original
  000135 1cfd                         adc mp10H,copyH
  000136 703f                         andi    adder,0x0f  ;mask away upper nibble of adder
  000137 0ee3                         add mp10L,adder ;add lower nibble of adder
```

```
000138 f408                         brcc    m10_1       ;if carry not cleared
000139 94f3                         inc mp10H       ;   inc high byte
00013a 9508                      m10_1:  ret

                                 ;***** Main Routine Register Variables

                                 .def    tbinL   =r14        ;Low byte of binary result (same as mp10L)
                                 .def    tbinH   =r15        ;High byte of binary result (same as mp10H)
                                 .def    fBCD0   =r16        ;BCD value digits 1 and 0
                                 .def    fBCD1   =r17        ;BCD value digits 2 and 3
                                 .def    fBCD2   =r18        ;BCD value digit 5

                                 ;***** Code

                                 BCD2bin16:
00013b 702f                         andi    fBCD2,0x0f  ;mask away upper nibble of fBCD2
00013c 24ff                         clr mp10H
00013d 2ee2                         mov mp10L,fBCD2 ;mp10H:mp10L = a
00013e 2f31                         mov adder,fBCD1
00013f dfe9                         rcall   mul10a      ;mp10H:mp10L = 10a+b
000140 2f31                         mov adder,fBCD1
000141 dfe8                         rcall   mul10b      ;mp10H:mp10L = 10(10a+b)+c
000142 2f30                         mov adder,fBCD0
000143 dfe5                         rcall   mul10a      ;mp10H:mp10L = 10(10(10a+b)+c)+d
000144 2f30                         mov adder,fBCD0
000145 dfe4                         rcall   mul10b      ;mp10H:mp10L = 10(10(10(10a+b)+c)+d)+e
000146 9508                         ret

                                 ;************************
                                 ;NAME:      clr_dsp_buffs
                                 ;FUNCTION:  Initializes dsp_buffers 1, 2, and 3 with blanks (0x20)
                                 ;ASSUMES:   Three CONTIGUOUS 16-byte dram based buffers named
                                 ;           dsp_buff_1, dsp_buff_2, dsp_buff_3.
                                 ;RETURNS:   nothing.
                                 ;MODIFIES:  r25,r26, Z-ptr
                                 ;CALLS:     none
                                 ;CALLED BY: main application and diagnostics
                                 ;****************************************************************** **
                                 clr_dsp_buffs:
000147 e390                           ldi R25, 48               ; load total length of both buffer.
000148 e2a0                           ldi R26, ' '              ; load blank/space into R26.
000149 e0f1                           ldi ZH, high (dsp_buff_1) ; Load ZH and ZL as a pointer to 1st
00014a e0ee                           ldi ZL, low (dsp_buff_1)  ; byte of buffer for line 1.

                                     ;set DDRAM address to 1st position of first line.
                                 store_bytes:
00014b 93a1                           st  Z+, R26       ; store ' ' into 1st/next buffer byte and
                                                        ; auto inc ptr to next location.
00014c 959a                           dec  R25          ;
00014d f7e9                           brne store_bytes  ; cont until r25=0, all bytes written.
00014e 9508                           ret
                                 ;****************************************************************** ****


                                 ;************************
                                 ;NAME:      prepare_for_BCD2bin
                                 ;FUNCTION:  prepares for the use of the BCD2bin function
                                 ;RETURNS:
                                 ;MODIFIES: Zh, ZL, r16, r17, r18
                                 ;cycles: 18 cycles
                                 ;words: 11 words
                                 ;(MSB)(MIDDLE DIGIT)(LSB)
                                 ; r16-------> (MIDDLE DIGIT)(LSB)
                                 ; r17-------> (0)(MSB)
                                 ; r18-------> (0)(0)
                                 ;****************************************************************** **
                                 prepare_for_BCD2bin:
                                 ;convert back into unpacked hex
00014f 5370                      subi r23, $30
000150 5360                      subi r22, $30
```

```
000151 5350                     subi r21, $30
                                ;storing it into the setting
000152 9371                     st Z+, r23
000153 9361                     st Z+, r22
000154 9351                     st Z+, r21

                                ;prepping for the BCD2BIN function -->43210
000155 9562                     swap r22
                                ;will give BCD digits 1 and 0
000156 2f05                     mov r16, r21
000157 0f06                     add r16, r22
                                ;will give BCD digits 3 and 2
000158 2f17                     mov r17, r23
                                ;set BCD digit 4 and 5 as 0
000159 e020                     ldi r18, $00
00015a 9508                     ret

                                ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;; ;;;;;;;;;
                                ;DELAY FUNCTIONS
                                ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;; ;;;;;;;;;


                                ;******************************************************************* *********
                                ;*
                                ;*Title: var_delay
                                ;* Description: Creates a delay for the ATMEGA324(has to run @ 1Mhz  CLK)
                                ;*
                                ;* Target:ATMEGA324A
                                ;* Number of words:6
                                ;* Number of cycles: 25248 cycles (when n = 32 and m = 255)
                                ;* n = inner loop variable and m = outer loop variable
                                ;* 3(nm+m+1)= total number of cycles
                                ;*
                                ;* High registers modified: r16, r17
                                ;* Parameters:r16
```

```
                                 ;*
                                 ;* Returns: N/A
                                 ;*
                                 ;*Delay provided should be around n*0.1ms
                                 ;*As stated before, n is the number inputted into r16
                                 ;***************************************************************** *********
                                 var_delay: ;delay for ATmega324 @ 1MHz = r16 * 0.1 ms
                                 outer_loop:
00015b e210                      ldi r17, 32
                                 inner_loop:
00015c 951a                      dec r17
00015d f7f1                      brne inner_loop
00015e 950a                      dec r16
00015f f7d9                      brne outer_loop
000160 9508                      ret


                                 ;*
                                 ;*Title: var_delay_2
                                 ;* Description: Creates a delay for the ATMEGA324(has to run @ 1Mhz  CLK)
                                 ;* Same purpose as var_delay, except that r17 is defined as 31
                                 ;* Target:ATMEGA324A
                                 ;* Number of words:6
                                 ;* Number of cycles:
                                 ;* n = inner loop variable and m = outer loop variable
                                 ;* 3(nm+m+1)= total number of cycles
                                 ;*
                                 ;* High registers modified: r16, r17
                                 ;* Parameters:r16
                                 ;*
                                 ;* Returns: N/A
                                 ;*
                                 ;*Delay provided should be around n*0.1ms
                                 ;*As stated before, n is the number inputted into r16
                                 ;***************************************************************** *********
```

```
                                var_delay_2:
                                outer:
000161 e11f                     ldi r17, 31
                                inner:
000162 951a                     dec r17
000163 f7f1                     brne inner
000164 950a                     dec r16
000165 f7d9                     brne outer
000166 9508                     ret

                                ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;; ;;;;;;;;;
                                ;PROMPTS AND UPDATES FOR THE PROMPTS
                                ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;; ;;;;;;;;;

                                ;****************************************************************** *********
                                ;*
                                ;*Title: prompt_burst_count
                                ;* Description: Creates n<space>=<space> on the LCD
                                ;* also used in order to set the X pointer pointing to the start of  the
                                ;*numbers to be displayed on the LCD
                                ;*
                                ;* Target:ATMEGA324A
                                ;* Number of words: 11 words
                                ;* Number of cycles: 6313 cycles
                                ;*
                                ;*
                                ;* High registers modified: r16, YH, YL
                                ;*
                                ;* Parameters:r16
                                ;*
                                ;* Returns: N/A
                                ;*
                                ;****************************************************************** *********
                                prompt_burst_count:
000167 e0d1                     ldi YH, high(dsp_buff_1)
```

```
000168 e0ce                       ldi YL, low(dsp_buff_1)

000169 e60e                       ldi r16, $6E ;displays n
00016a 9309                       st Y+, r16

00016b e200                       ldi r16, $20;displays <space>
00016c 9309                       st Y+, r16

00016d e30d                       ldi r16, $3D ;displays =
00016e 9309                       st Y+, r16

00016f e200                       ldi r16, $20 ;displays <space>
000170 9309                       st Y+, r16
000171 df3a                       rcall update_lcd_dog
                                  ;at the end, should display n<space>=<space>
                                  ;on the LCD
000172 9508                       ret

                                  ;****************************************************************** *********
                                  ;*
                                  ;*Title: update_burst_count
                                  ;* Description: meant to update the LCD buffer when doing the burst  prompt
                                  ;*in real time (for the burst count line (line 1))
                                  ;*
                                  ;* Target:ATMEGA324A
                                  ;* Number of words: 5 words
                                  ;* Number of cycles: 6326 cycles
                                  ;*
                                  ;* High registers modified: YH, YL
                                  ;* Parameters:r16
                                  ;*
                                  ;* Returns: N/A
                                  ;*
                                  ;*calls prompt burst count to reinitialize that line for the n =
                                  ;****************************************************************** *********
```

```

                                   update_burst_count:
000173 9379                        st Y+, r23
000174 9369                        st Y+, r22
000175 9359                        st Y+, r21
000176 dff0                        rcall prompt_burst_count
000177 9508                        ret


                                   ;****************************************************************** *********
                                   ;*
                                   ;*Title: prompt_pulse_width_count
                                   ;* Description: Creates t<space>=<space> on the LCD
                                   ;* also used in order to set the X pointer pointing to the start of  the
                                   ;*numbers to be displayed on the LCD
                                   ;*
                                   ;* Target:ATMEGA324A
                                   ;* Number of words: 11 words
                                   ;* Number of cycles: 6313 cycles
                                   ;*
                                   ;*
                                   ;* High registers modified: r16, YH, YL
                                   ;*
                                   ;* Parameters:r16
                                   ;*
                                   ;* Returns: N/A
                                   ;*
                                   ;****************************************************************** *********
                                   prompt_pulse_width_count:
000178 e0d1                        ldi YH, high(dsp_buff_2)
000179 e1ce                        ldi YL, low(dsp_buff_2)

00017a e704                        ldi r16, 't' ;displays t
00017b 9309                        st Y+, r16

```

```
00017c e200                       ldi r16, $20;displays <space>
00017d 9309                       st Y+, r16

00017e e30d                       ldi r16, $3D ;displays =
00017f 9309                       st Y+, r16

000180 e200                       ldi r16, $20 ;displays <space>
000181 9309                       st Y+, r16
000182 df29                       rcall update_lcd_dog
                                  ;at the end, should display t<space>=<space>
                                  ;on the LCD
000183 9508                       ret


                                  ;***************************************************************** *********
                                  ;*
                                  ;*Title: update_pulse_width
                                  ;* Description: meant to update the LCD buffer when doing the pulse  width
                                  ;*in real time (for the pulse width line (line 2))
                                  ;*
                                  ;* Target:ATMEGA324A
                                  ;* Number of words: 5 words
                                  ;* Number of cycles: 6326 cycles
                                  ;*
                                  ;* High registers modified: YH, YL
                                  ;*
                                  ;* Returns: N/A
                                  ;*
                                  ;*calls prompt pulse width count to reinitialize that line for the  t =
                                  ;***************************************************************** *********

                                  update_pulse_width:
000184 9379                       st Y+, r23
000185 9369                       st Y+, r22
000186 9359                       st Y+, r21
```

```
000187 dff0                     rcall prompt_pulse_width_count
000188 9508                     ret

                                ;******************************************************************* *********
                                ;*
                                ;*Title: prompt_delay_count
                                ;* Description: Creates d<space>=<space> on the LCD
                                ;* also used in order to set the X pointer pointing to the start of  the
                                ;*numbers to be displayed on the LCD
                                ;*
                                ;* Target:ATMEGA324A
                                ;* Number of words: 11 words
                                ;* Number of cycles: 6313 cycles
                                ;*
                                ;*
                                ;* High registers modified: r16, YH, YL
                                ;*
                                ;* Parameters:r16
                                ;*
                                ;* Returns: N/A
                                ;*
                                ;******************************************************************* *********
                                prompt_delay_count:
000189 e0d1                     ldi YH, high(dsp_buff_3)
00018a e2ce                     ldi YL, low(dsp_buff_3)

00018b e604                     ldi r16, 'd' ;displays n
00018c 9309                     st Y+, r16

00018d e200                     ldi r16, $20;displays <space>
00018e 9309                     st Y+, r16

00018f e30d                     ldi r16, $3D ;displays =
000190 9309                     st Y+, r16

```

```
000191 e200                      ldi r16, $20 ;displays <space>
000192 9309                      st Y+, r16
000193 df18                      rcall update_lcd_dog
                                 ;at the end, should display n<space>=<space>
                                 ;on the LCD
000194 9508                      ret

                                 ;****************************************************************** *********
                                 ;*
                                 ;*Title: update_delay
                                 ;* Description: meant to update the LCD buffer when doing the pulse  width
                                 ;*in real time
                                 ;*
                                 ;* Target:ATMEGA324A
                                 ;* Number of words: 5 words
                                 ;* Number of cycles: 6326 cycles
                                 ;*
                                 ;* High registers modified: r23,r22, r21, YH, YL
                                 ;*
                                 ;* Returns: N/A
                                 ;*
                                 ;*calls prompt burst count to reinitialize that line for the t =
                                 ;****************************************************************** *********

                                 update_delay:
000195 9379                      st Y+, r23
000196 9369                      st Y+, r22
000197 9359                      st Y+, r21
000198 dff0                      rcall prompt_delay_count
000199 9508                      ret

                                 ;****************************************************************** *********
                                 ;*
                                 ;* "fsm" - Simplified Table Driven Finite State Machine
                                 ;*
```

```
;* Description:
;* This table driven FSM can handle 255 or fewer input symbols.
;*
;* Author:              Ken Short
;* Version:             2.0
;* Last updated:        11/09/15
;* Target:              ATmega16
;* Number of words:
;* Number of cycles:
;* Low regs modified:   r16, r18, r20, r21, r31, and r31
;* High registers used:
;*
;* Parameters:          present state in r25:r24 prior to call
;*                      input symbol in r16 prior to call
;*
;* Notes:
;*
;****************************************************************** *********

;input symbols for example finite state machine
.equ i0 = $00   ;input symbols equated to numerical values ;
.equ i1 = $01
.equ i2 = $02
.equ i3 = $03
.equ i4 = $04
.equ i5 = $05
.equ i6 = $06
.equ i7 = $07
.equ i8 = $08
.equ i9 = $09
.equ UP = $0F
.equ DOWN = $0E
.equ key2nd = $0D
.equ CLEAR = $0A
.equ HELP = $0B
```

```
                                  .equ ENTERED = $0C
                                  .equ pbpress = $10
                                  .equ eol = $FF  ;end of list (subtable) do not change

                                  ;state table for example finite state machine
                                  ;each row consists of input symbol, next state address, task
                                  ;subroutine address

                                  state_table:

                                  clr_screen:
00019a 000a
00019b 01a0
00019c 01ef                           .dw CLEAR,     input_burst,        task1
00019d 00ff
00019e 019a
00019f 01e7                           .dw eol,   clr_screen,     task0

                                  input_burst:
0001a0 000c
0001a1 01be
0001a2 026f                          .dw ENTERED,        check_trigger,      task8
0001a3 000e
0001a4 01a9
0001a5 021b                           .dw DOWN,      input_width,        task3
0001a6 00ff
0001a7 01a0
0001a8 0200                           .dw eol,   input_burst,        task2

                                  input_width:
0001a9 000c
0001aa 01be
0001ab 026f                          .dw ENTERED,        check_trigger,      task8
0001ac 000f
0001ad 01a0
```

```
0001ae 0249                          .dw UP,             input_burst,        task6
0001af 000e
0001b0 01b5
0001b1 023f                         .dw DOWN,       input_delay,        task5
0001b2 00ff
0001b3 01a9
0001b4 0225                          .dw eol,    input_width,        task4
                                 input_delay:
0001b5 000c
0001b6 01be
0001b7 026f                         .dw ENTERED,        check_trigger,      task8
0001b8 000f
0001b9 01a9
0001ba 021b                          .dw UP,         input_width,        task3
0001bb 00ff
0001bc 01b5
0001bd 0253                          .dw eol,    input_delay,        task7

                                 check_trigger:
0001be 0010
0001bf 0000
0001c0 02ac                         .dw pbpress,        0,      task11
0001c1 00ff
0001c2 01be
0001c3 02a7                          .dw eol,    check_trigger,      task10

                                 normal_burst_int_trigger:
0001c4 0010
0001c5 01c4
0001c6 02a0                         .dw pbpress,        normal_burst_int_trigger,       task9
0001c7 000a
0001c8 01a0
0001c9 01ef                          .dw CLEAR,      input_burst,        task1
0001ca 00ff
0001cb 01c4
```

```
0001cc 02a7                         .dw eol,    normal_burst_int_trigger,       task10

                                continuous_chk_clr:
0001cd 000a
0001ce 01a0
0001cf 01ef                        .dw CLEAR,      input_burst,        task1
0001d0 00ff
0001d1 01cd
0001d2 02a7                         .dw eol,    continuous_chk_clr,     task10


                                fsm:
                                ;load Z with a byte pointer to the subtable corresponding to the
                                ;present state
0001d3 2fe8                         mov ZL, pstatel ;load Z pointer with pstate address * 2
0001d4 0fee                         add ZL, ZL ;since Z will be used as a byte pointer with the lpm  instr.
0001d5 2ff9                         mov ZH, pstateh
0001d6 1fff                         adc ZH, ZH

                                ;search subtable rows for input symbol match
                                search:
0001d7 9124                         lpm r18, Z ;get symbol from state table
0001d8 1720                         cp r18, r16 ;compare table entry with input symbol
0001d9 f021                         breq match

                                ;check input symbol against eol
                                check_eol:
0001da 3f2f                         cpi r18, eol ;compare low byte of table entry with eol
0001db f011                         breq match

                                nomatch:
0001dc 9636                         adiw ZL, $06 ;adjust Z to point to next row of state table
0001dd cff9                         rjmp search ;continue searching

                                ;a match on input value to row input value has been found
                                ;the next word in this row is the next state address
                                ;the word following that is the task subroutine's address
                                match:
                                   ;make preseent state equal to next state value in row
                                   ;this accomplishes the stat transition
0001de 9632                         adiw ZL, $02 ;point to low byte of state address
0001df 9185                         lpm pstatel, Z+; ;copy next state addr. from table to preseent  stat
0001e0 9195                         lpm pstateh, Z+

                                   ;execute the subroutine that accomplihes the task associated
                                   ;with the transition
0001e1 9135                         lpm r19, Z+ ;get subroutine address from state table
0001e2 9144                         lpm r20, Z ;and put it in Z pointer
0001e3 2fe3                         mov ZL, r19
0001e4 2ff4                         mov ZH, r20
0001e5 9509                         icall ;Z pointer is now used as a word pointer
0001e6 9508                         ret



                                ;******************************************************************* *********
                                ;*
                                ;* "taskn" - Stub subroutines for testing
                                ;*
                                ;* Description:
                                ;* These subroutines are the tasks for the simple table driven FSM  example.
                                ;* When a program is being developed, you should start with each of  these
                                ;* subroutines consisting of just a nop and a return. You can then  simulate
                                ;* the program and verify that the transitions defined by you trans ition
                                ;* table and original state diagram take place in response to input
                                ;* sequences.
                                ;*
                                ;* Author:              Ken Short
                                ;* Version:
                                ;* Last updated:
                                ;* Target:
                                ;* Number of words:
                                ;* Number of cycles:
                                ;* Low registers used:
                                ;* High registers used:
                                ;*
                                ;* Parameters:
                                ;*
                                ;* Notes:
                                ;*
                                ;****************************************************************** *********

                                ;sounds the buzzer for all inputs not CLR, and clears the screen
                                task0:
0001e7 df5f                        rcall clr_dsp_buffs;displays a blank screen
                                ;rcall update_lcd_dog;updates the screen

                                ;put FSM in initial state
0001e8 e98a                        ldi pstatel,low(clr_screen)
0001e9 e091                        ldi pstateh, high(clr_screen)

                                ;sounds the buzzer
0001ea ef0f                        ldi r16, 255
0001eb 9a16                        sbi PORTA, 6
0001ec df6e                        rcall var_delay
0001ed 9816                        cbi PORTA, 6
0001ee 9508                        ret

                                ;displays all prompts, makes burst line active
                                task1:
                                ;clears all flags
0001ef e002                        ldi r16, 2
0001f0 9300 010d                   sts normal_flag, r16

0001f2 e002                        ldi r16, 2
```

```
0001f3 9300 010c                 sts continuous_flag, r16
                                 ;initializes all lines to 0

                                 ;initial values for the burst count
0001f5 e350                      ldi r21, $30
0001f6 e360                      ldi r22, $30
0001f7 e370                      ldi r23, $30

                                 ;setting r21,r20 and r19 to zero in case of enter
                                 ;pressed without inputting values
0001f8 df6e                      rcall prompt_burst_count
0001f9 df79                      rcall update_burst_count
0001fa df03                      rcall select_line

0001fb df7c                      rcall prompt_pulse_width_count
0001fc df87                      rcall update_pulse_width

0001fd df8b                      rcall prompt_delay_count
0001fe df96                      rcall update_delay

0001ff 9508                      ret

                                 ;inputs digits into burst buffer, sounds buzzer for keys not used
                                 task2:
                                 ;-------------------------------------------------
                                 ;GETTING BURST COUNT
                                 ;-------------------------------------------------
                                 ;getting the value of the burst count
                                 ;meant to run in an infinite loop getting values
                                 ;until enter is pressed
                                 ;r23-->MSB
                                 ;r21-->LSB

                                 input1: ;corresponding to line 1 of the LCD

```

```
000200 2f20                      mov r18, r16
                                 ;if CLR is pressed, take another input instead
000201 302a                      cpi r18, $0A
000202 f099                      breq endwithbuzzer

                                 ;if pushbutton is pressed, take another input
000203 3120                      cpi r18, pbpress
000204 f089                      breq endwithbuzzer

                                 ;if up arrow is pressed, ignore input
000205 302f                      cpi r18, $0F
000206 f079                      breq endwithbuzzer

000207 df0f                      rcall check_unneeded_buttons
000208 f069                      breq endwithbuzzer

000209 2f76                      mov r23, r22
00020a 2f65                      mov r22, r21

00020b 2f52                      mov r21, r18

00020c df10                      rcall convert_hex
00020d df59                      rcall prompt_burst_count
00020e df64                      rcall update_burst_count
00020f deee                      rcall select_line
000210 df67                      rcall prompt_pulse_width_count
000211 deef                      rcall deselect_line
000212 df76                      rcall prompt_delay_count
000213 deed                      rcall deselect_line
000214 de97                      rcall update_lcd_dog
000215 9508                      ret

                                 ;sounds the buzzer for useless inputs
                                 endwithbuzzer:
000216 ef0f                      ldi r16, 255
```

```
000217 9a16                    sbi PORTA, 6
000218 df42                    rcall var_delay
000219 9816                    cbi PORTA, 6
00021a 9508                    ret


                               ;width line active (makes other lines inactive)
                               task3:
00021b df5c                    rcall prompt_pulse_width_count
00021c def6                    rcall keep_values
00021d df5a                    rcall prompt_pulse_width_count
00021e dedf                    rcall select_line
00021f df47                    rcall prompt_burst_count
000220 dee0                    rcall deselect_line
000221 df67                    rcall prompt_delay_count
000222 dede                    rcall deselect_line
000223 de88                    rcall update_lcd_dog
000224 9508                    ret

                               ;inputs digits into width buffer, sounds buzzer for keys not used
                               task4:
                               ;-------------------------------------------------
                               ;GETTING PULSE WIDTH
                               ;-------------------------------------------------

                               input2:
000225 2f20                    mov r18, r16
                               ;if CLR is pressed, take another input instead
000226 302a                    cpi r18, $0A
000227 f091                    breq endwithbuzzer2

                               ;if pushbutton is pressed, take another input
000228 3120                    cpi r18, pbpress
000229 f081                    breq endwithbuzzer2

```

```
00022a deec                         rcall check_unneeded_buttons
00022b f071                         breq endwithbuzzer2

00022c 2f76                         mov r23, r22
00022d 2f65                         mov r22, r21

00022e 2f52                         mov r21, r18


00022f 2f52                         mov r21, r18
000230 deec                         rcall convert_hex
000231 df35                         rcall prompt_burst_count
000232 dece                         rcall deselect_line
000233 df55                         rcall prompt_delay_count
000234 decc                         rcall deselect_line
000235 df42                         rcall prompt_pulse_width_count
000236 df4d                         rcall update_pulse_width
000237 dec6                         rcall select_line
000238 de73                         rcall update_lcd_dog

000239 9508                         ret

                                    ;sounds the buzzer for useless inputs
                                    endwithbuzzer2:
00023a ef0f                         ldi r16, 255
00023b 9a16                         sbi PORTA, 6
00023c df1e                         rcall var_delay
00023d 9816                         cbi PORTA, 6
00023e 9508                         ret

                                    ;delay line active
                                    task5:
                                    inner_loop3:
00023f df49                         rcall prompt_delay_count
000240 ded2                         rcall keep_values
```

```
000241 df47                     rcall prompt_delay_count
000242 debb                     rcall select_line
000243 df23                     rcall prompt_burst_count
000244 debc                     rcall deselect_line
000245 df32                     rcall prompt_pulse_width_count
000246 deba                     rcall deselect_line
000247 de64                     rcall update_lcd_dog
000248 9508                     ret

                                ;burst line active
                                task6:
000249 df1d                     rcall prompt_burst_count
00024a dec8                     rcall keep_values
00024b df1b                     rcall prompt_burst_count
00024c deb1                     rcall select_line
00024d df2a                     rcall prompt_pulse_width_count
00024e deb2                     rcall deselect_line
00024f df39                     rcall prompt_delay_count
000250 deb0                     rcall deselect_line
000251 de5a                     rcall update_lcd_dog
000252 9508                     ret

                                ;inputs digits into delay buffer, sounds buzzer for keys not used
                                task7:
                                ;------------------------------------------------
                                ;GETTING DELAY COUNT
                                ;------------------------------------------------

                                input3:;corresponding to line 3 of the LCD
000253 2f20                     mov r18, r16
                                ;if CLR is pressed, take another input instead
000254 302a                     cpi r18, $0A
000255 f0a1                     breq endwithbuzzer3

                                ;if pushbutton is pressed, take another input
```

```
000256 3120                       cpi r18, pbpress
000257 f091                       breq endwithbuzzer3

                                  ;if down arrow is pressed, read another input again
000258 302e                       cpi r18, $0E
000259 f081                       breq endwithbuzzer3

00025a debc                       rcall check_unneeded_buttons
00025b f071                       breq endwithbuzzer3

00025c 2f76                       mov r23, r22
00025d 2f65                       mov r22, r21

00025e 2f52                       mov r21, r18


00025f 2f52                       mov r21, r18
000260 debc                       rcall convert_hex
000261 df05                       rcall prompt_burst_count
000262 de9e                       rcall deselect_line
000263 df14                       rcall prompt_pulse_width_count
000264 de9c                       rcall deselect_line
000265 df23                       rcall prompt_delay_count
000266 df2e                       rcall update_delay
000267 de96                       rcall select_line
000268 de43                       rcall update_lcd_dog
000269 9508                       ret
                                  ;sounds the buzzer for useless inputs
                                  endwithbuzzer3:
00026a ef0f                       ldi r16, 255
00026b 9a16                       sbi PORTA, 6
00026c deee                       rcall var_delay
00026d 9816                       cbi PORTA, 6
00026e 9508                       ret
```

```
                                ;saves all settings, if enter is pressed
                                task8:
                                ;this is for inputting the prompt burst count into memory
00026f def7                     rcall prompt_burst_count
000270 de90                     rcall deselect_line


000271 def5                     rcall prompt_burst_count
000272 9179                     ld r23, Y+
000273 9169                     ld r22, Y+
000274 9159                     ld r21, Y+

                                ;storing burst_count_bcd_setting
000275 e0f1                     ldi ZH, high(burst_count_bcd_setting)
000276 e0e0                     ldi ZL, low(burst_count_bcd_setting)

                                ;prepares for the BCD2bin
                                ;stores teh bcd setting into SRAM and
                                ;gets ready to input to the BCD2bin function
000277 ded7                     rcall prepare_for_BCD2bin

000278 dec2                     rcall BCD2bin16

000279 e0d1                     ldi YH, high(burst_count_binary_setting)
00027a e0c3                     ldi YL, low(burst_count_binary_setting)
                                ;has the value of binary setting stored in Y
00027b 82e8                     st Y, r14

                                ;this is for storing the pulse width
                                ;this is for inputting the prompt burst count into memory

00027c defb                     rcall prompt_pulse_width_count
00027d de83                     rcall deselect_line

00027e def9                     rcall prompt_pulse_width_count
```

```
00027f 9179                 ld r23, Y+
000280 9169                 ld r22, Y+
000281 9159                 ld r21, Y+

                            ;storing burst_count_bcd_setting
000282 e0f1                 ldi ZH, high(pulse_width_bcd_setting)
000283 e0e4                 ldi ZL, low(pulse_width_bcd_setting)


000284 deca                 rcall prepare_for_BCD2bin

000285 deb5                 rcall BCD2bin16

000286 e0d1                 ldi YH, high(pulse_width_binary_setting)
000287 e0c7                 ldi YL, low(pulse_width_binary_setting)
                            ;has the value of binary setting stored in Y
000288 82e8                 st Y, r14

                            ;this is for inputting the prompt delay count into memory
000289 deff                 rcall prompt_delay_count
00028a de76                 rcall deselect_line

00028b defd                 rcall prompt_delay_count
00028c 9179                 ld r23, Y+
00028d 9169                 ld r22, Y+
00028e 9159                 ld r21, Y+

                            ;storing delay_time_bcd_setting
00028f e0f1                 ldi ZH, high(delay_time_bcd_setting)
000290 e0e8                 ldi ZL, low(delay_time_bcd_setting)

                            ;prepares for the BCD2bin
                            ;stores teh bcd setting into SRAM and
                            ;gets ready to input to the BCD2bin function
000291 debd                 rcall prepare_for_BCD2bin
```

```

000292 dea8                     rcall BCD2bin16

000293 e0d1                     ldi YH, high(delay_time_binary_setting)
000294 e0cb                     ldi YL, low(delay_time_binary_setting)
                                ;has the value of binary setting stored in Y
000295 82e8                     st Y, r14


                                ;taking the burst count setting
000296 e0d1                     ldi YH, high(burst_count_binary_setting)
000297 e0c3                     ldi YL, low(burst_count_binary_setting)
000298 80e8                     ld r14, Y
000299 2d6e                     mov r22, r14

                                ;taking the pulse width count setting
00029a e0d1                     ldi YH, high(pulse_width_binary_setting)
00029b e0c7                     ldi YL, low(pulse_width_binary_setting)
00029c 80d8                     ld r13, Y

                                ;taking the delay count setting
00029d e0d1                     ldi YH, high(delay_time_binary_setting)
00029e e0cb                     ldi YL, low(delay_time_binary_setting)
00029f 9508                     ret

                                ;reinit burst setting
                                task9:
0002a0 e0d1                     ldi YH, high(burst_count_binary_setting)
0002a1 e0c3                     ldi YL, low(burst_count_binary_setting)
0002a2 8168                     ld r22, Y

                                ;3 means that you redo the burst setting
0002a3 e003                     ldi r16, 3
0002a4 9300 010d                sts normal_flag, r16
0002a6 9508                     ret
```

```


                                  ;sounds buzzer for all inputs not CLR or pbpress
                                  task10:
0002a7 ef0f                       ldi r16, 255
0002a8 9a16                       sbi PORTA, 6
0002a9 deb1                       rcall var_delay
0002aa 9816                       cbi PORTA, 6
0002ab 9508                       ret

                                  ;check to see if the burst count is 0 or a number
                                  task11:
0002ac 90e0 0103                  lds r14, burst_count_binary_setting
0002ae 2d6e                       mov r22, r14
0002af 3060                       cpi r22, 0
0002b0 f049                       breq continuous_flag_set

                                  normal_flag_set:
0002b1 e001                       ldi r16,1
0002b2 9300 010d                  sts normal_flag, r16
0002b4 e000                       ldi r16, 0
0002b5 9300 010c                  sts continuous_flag, r16
0002b7 ec84                       ldi pstatel, low(normal_burst_int_trigger)
0002b8 e091                       ldi pstateh, high(normal_burst_int_trigger)
0002b9 9508                       ret

                                  continuous_flag_set:
0002ba e001                       ldi r16,1
0002bb 9300 010c                  sts continuous_flag, r16
0002bd e000                       ldi r16,0
0002be 9300 010d                  sts normal_flag, r16
0002c0 ec8d                       ldi pstatel, low(continuous_chk_clr)
0002c1 e091                       ldi pstateh, high(continuous_chk_clr)
0002c2 9508                       ret

```

```




RESOURCE USE INFORMATION
------------------------

Notice:
The register and instruction counts are symbol table hit counts,
and hence implicitly used resources are not counted, eg, the
'lpm' instruction without operands implicitly uses r0 and z,
none of which are counted.

x,y,z are separate entities in the symbol table and are
counted separately from r26..r31 here.

.dseg memory usage only counts static data declared with .byte

"ATmega324A" register use summary:
x  :   0 y  :  45 z  :  13 r0 :   0 r1 :   0 r2 :   0 r3 :   0 r4 :   0
r5 :   0 r6 :   0 r7 :   0 r8 :   0 r9 :   0 r10:   0 r11:   0 r12:   5
r13:  10 r14:  12 r15:   9 r16: 153 r17:  10 r18:  33 r19:   9 r20:  10
r21:  20 r22:  25 r23:  16 r24:  10 r25:   8 r26:   2 r27:   0 r28:  14
r29:  14 r30:  15 r31:  13
Registers used: 21 out of 35 (60.0%)

"ATmega324A" instruction use summary:
.lds  :   0 .sts  :   0 adc   :   3 add   :   6 adiw  :   2 and   :   0
andi  :   4 asr   :   0 bclr  :   0 bld   :   0 brbc  :   0 brbs  :   0
brcc  :   1 brcs  :   0 break :   0 breq  :  21 brge  :   0 brhc  :   0
brhs  :   0 brid  :   0 brie  :   0 brlo  :   0 brlt  :   0 brmi  :   0
brne  :  13 brpl  :   0 brsh  :   0 brtc  :   0 brts  :   0 brvc  :   0
brvs  :   0 bset  :   0 bst   :   0 call  :   0 cbi   :  12 cbr   :   0
clc   :   2 clh   :   0 cli   :   0 cln   :   0 clr   :   1 cls   :   0
clt   :   0 clv   :   0 clz   :   0 com   :   0 cp    :   1 cpc   :   0
```

```
cpi   :  17 cpse  :   0 dec   :  12 eor   :   0 fmul  :   0 fmuls :   0
fmulsu:   0 icall :   1 ijmp  :   0 in    :  13 inc   :   1 jmp   :   2
ld    :  22 ldd   :   0 ldi   : 121 lds   :   6 lpm   :   9 lsl   :   4
lsr   :   0 mov   :  36 movw  :   0 mul   :   0 muls  :   0 mulsu :   0
neg   :   0 nop   :   4 or    :   0 ori   :   0 out   :  14 pop   :  10
push  :  10 rcall : 144 ret   :  47 reti  :   2 rjmp  :   9 rol   :   4
ror   :   0 sbc   :   0 sbci  :   0 sbi   :  16 sbic  :   1 sbis  :   3
sbiw  :   0 sbr   :   0 sbrc  :   0 sbrs  :   2 sec   :   2 seh   :   0
sei   :   1 sen   :   0 ser   :   0 ses   :   0 set   :   0 sev   :   0
sez   :   0 sleep :   0 spm   :   0 st    :  28 std   :   2 sts   :  10
sub   :   0 subi  :   3 swap  :   4 tst   :   0 wdr   :   0
Instructions used: 43 out of 113 (38.1%)

"ATmega324A" memory use summary [bytes]:
Segment   Begin    End      Code   Data   Used    Size   Use%
---------------------------------------------------------------
[.cseg] 0x000000 0x000586   1282    130   1412   32768   4.3%
[.dseg] 0x000100 0x00013e      0     62     62    2048   3.0%
[.eseg] 0x000000 0x000000      0      0      0    1024   0.0%

Assembly complete, 0 errors, 2 warnings

```