# **ATMEL Studio 7 (main.asm)**

I’ve organized this code presentation into **4** sections, hopefully making it clearer:  
 Page 1. **Directives**, Page 2. **Digit Separation**, Page 3. **PoV Display**, Page 4. **Shiftout** followed by the delay code from the [Javascript utility](http://darcy.rsgc.on.ca/ACES/TEI4M/AVRdelay.html).

;PROJECT :POVontheADCShield

;PURPOSE :POV of 2-digit display on ADC Shield

;AUTHOR :C. D'Arcy

;DATE :2020 04 24

;DEVICE :Dolgin Development Platform

;MCU :ATtiny84

;COURSE :ICS4U

;STATUS :Working

;REFERENCE :https://mail.rsgc.on.ca/~cdarcy/Datasheets/doc0856.pdf

.def count = r15 ;accumulator for the multiples...

.def util = r16 ;readability is enhanced through aliases

.def value = r17 ;holds the working value to be displayed

.def mask = r18 ;byte with one set bit to act as a mask

.def dir = r19 ;holds shift direction: LSBFIRST:0, MSBFIRST:1

.def n = r20 ;holds the number being shifted out

.equ DDR = DDRA ;typically, we'll need the use of PortA

.equ PORT = PORTA ;both its data direction register and output register

;.equ thous = PA1 ;these are the port pins...

;.equ hunds = PA2 ; connected to the base pins of

.equ tens = PA3 ; each of the transistors that

.equ units = PA4 ; ground the respective displays

.equ DATA = PA5 ;595 data pin

.equ LATCH = PA6 ;595 latch pin

.equ CLOCK = PA7 ;595 clock pin

;.equ LSBFIRST= 0 ;same familiar constants from Arduino days

.equ MSBFIRST= 1 ;ditto

.dseg

.org SRAM\_START

segments:

.BYTE 2 ;space reserved for the separated digits

.cseg ; ; FLASH (Code) Segment

; \*\*\*\*\* INTERRUPT VECTOR TABLE \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

.org 0x0000 ;start of Interrupt Vector Table (IVT) aka. Jump Table

rjmp reset ;lowest interrupt address == highest priority!

.org 0x0020 ;position segment LUT beyond the IVT

segStart: ;MSBFIRST: ABCDEFGx

.db 0b11111100,0b01100000,0b11011010,0b11110010 ;0-3

.db 0b01100110,0b10110110,0b10111110,0b11100000 ;4-7

.db 0b11111110,0b11110110,0b11101110,0b00111110 ;8-b

.db 0b10011100,0b01111010,0b10011110,0b10001110 ;C-F

segEnd:

number:

.db 99,0 ;hard code the number to be displayed

; \*\*\*\*\* START OF CODE \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

.org 0x0100 ;safe standard address for start of executable code

reset: ;PC jumps to here (start of code) on reset interrupt...

ldi ZL,low(number<<1) ;obtain a pointer to the 2-digit number to

ldi ZH,high(number<<1) ; be presented on the displays

lpm util,Z ;obtain the number to be separated

clr count ;zero the count for the tens digit

dig10: ;division accomplished by repeated subtractions

subi util,10 ;subtract 10

brmi tDone ;are we negative?

inc count ;if not, add 1 to count

rjmp dig10 ;repeat

tDone: ;tens digit identified; repeated subtractions finished

sts segments,count ;the count is the tens digit: store in SRAM

subi util,-10 ;add 10 to get back above 0

sts segments+1,util ;store the units digit in SRAM

; \*\*\*\* PRESENT the TWO DIGITS on the DISPLAYS \*\*\*\*

ser util ;just as efficient to set all bits for output

out DDR,util ;set 'em

ldi dir,MSBFIRST ;this application is defined for MSBFIRST

repeat:

ldi ZL,low(segStart<<1) ;establish pointer to the base of the

ldi ZH,high(segStart<<1) ; 10-cell segment array

lds util,segments+1 ;read the units digit from SRAM

add ZL,util ;use it as an offset from the base of the array

lpm n,Z ;load it into the required register for shifting

cbi PORT,tens ;turn off the currently displayed tens digit

rcall shiftout ;shift into the 595's internal registers and present

sbi PORT,units ;turn on the units digits display to see it

rcall delay10ms ;admire for 10ms or 100Hz (higher rate than PoV)

ldi ZL,low(segStart<<1) ;establish pointer to the base of the

ldi ZH,high(segStart<<1) ; 10-cell segment array

lds util,segments ;read the tens digit from SRAM

add ZL,util ;use it as an offset from the base of the array

lpm n,Z ;load it into the required register for shifting

cbi PORT,units ;turn off the currently displayed units digit

rcall shiftout ;shift into the 595's internal registers and present

sbi PORT,tens ;turn on the tens digits display to see it

rcall delay10ms ;admire for 10ms or 100Hz (higher rate than PoV)

rjmp repeat ;do it again...

; \*\*\*\* SHIFTOUT \*\*\*\*

shiftout: ;shifts constant n into the 595

ldi mask,0x80 ;assume order is MSBFIRST

sbrs dir,0 ;if bit 0 is set, it's MSBFIRST

ldi mask,0x01 ;OK, it's LSBFIRST so redefine the mask

cbi PORT,LATCH ;pull LATCH pin LOW

again:

cbi PORT,CLOCK ;pull CLOCK pin LOW

mov value,n ;reload the value to be presented

and value,mask ;mask off the target bit

breq lo ;was it 0?

sbi PORT,DATA ;no, so pull DATA pin HIGH

rjmp clockit ;ready to clock the 1

lo:

cbi PORT,DATA ;else, it was a 0, so pull DATA pin LOW

clockit:

sbi PORT,CLOCK ;pull CLOCK pin HIGH

; hmmm, must decide what direction to shift the mask...

sbrs dir,0 ;if bit 0 is set, it's MSBFIRST

rjmp shiftLeft ;OK, it's LSBFIRST

lsr mask ;MSBFIRST so shift the mask right

brne again ;repeat if there are still more bits to stuff in

rjmp done ;we're done, so only one more thing to do

shiftLeft:

lsl mask ;LSBFIRST, so shift the mask left

brne again ;repeat if there are still more bits to stuff in

done:

sbi PORT,LATCH ;pull LATCH pin HIGH to present 595's internal latches on output pins

ret ;finished, return.

; \*\*\*\* DELAY \*\*\*\*

delay10ms:

; Assembly code auto-generated

; by utility from Bret Mulvey

; Delay 80 000 cycles

; 10ms at 8.0 MHz

ldi r21, 104

ldi r22, 229

L1: dec r22

brne L1

dec r21

brne L1

ret