Part3a)

jmp Init ; first line executed

Jmp InterRout0

Jmp InterRout1

.org 0x0030

;=========

Init:

; initialize Stack Pointer (SP)

ldi r16, LOW(RAMEND)

out SPL, r16

ldi r16, HIGH(RAMEND)

out SPH, r16

; initialize Port B

ser r16 ; set all PortB pins to output

out DDRB, r16 ;

; initialize PortA

ldi r16, 0b11110000 ; set PA4-PA7 as outputs, PA0-PA3 as inputs

out DDRA, r16

ldi r16, 0b11111111 ; enable pull-up registers on PA4-PA7

out PORTA, r16

;=================

; Main body of program

Start:

ser r25 ; initialize LEDs to off

out PORTB , r25

ldi r21, 1; initializes the number of delay calls to 1 (range of 1 to 10)

StartIn:

ldi YL, low(TextString)

ldi YH, high(TextString)

ldi R17, 0; Register to track string length

Checkpress1:

Ldi ZL, low(2\*TextArray) ;R30

Ldi ZH, high(2\*TextArray); R31

call GetKeyPress ; check for a keypress

mov r24, r0 ; if key not pressed, check again

cpi r24, 0

breq Checkpress1

mov r25, r1 ; if key pressed, update display

ldi r16, 4

Mul r16,r25 ; multiply the button pos by 4

Add r30, r0;

Adc r31, r1; Add that value to the memory location of the text array

Release1:

call GetKeyPress ; check for button release

mov r24, r0 ; if key not released, check again

cpi r24, 0

brne Release1

Checkpress2:

call GetKeyPress ; check for a keypress

mov r24, r0 ; if key not pressed, check again

cpi r24, 0

breq Checkpress2

mov r25, r1 ; if key pressed, update display

clr r16

lsr r25

lsr r25 ; A = 0, D = 3

add r30, r25;

Adc R31, r16;

Lpm R25, Z;

Release2:

call GetKeyPress ; check for button release

mov r24, r0 ; if key not released, check again

cpi r24, 0

brne Release2

cpi r25, 0x23;

breq Display; If the input matches #, then move to displaying

com r25

st Y+, r25 ; Stores the inverted char selected in the string array

inc R17; Increment the string length counter

rjmp Checkpress1; repeat until # is entered

Display:

ldi YL, low(TextString)

ldi YH, high(TextString) ; reset the string index

ld r25, Y+

out PORTB, r25 ; Displays the char selected

mov r16, r21

DelLoop:

call MyDelay

dec r16

brne DelLoop

dec r17

brne display

rjmp StartIn; Reset to accepting input

End:

rjmp End ; end of program

;=================

; Procedures

;Procedure: GetKeyPress

; Function: checks whether a key has been pressed, and if so, which button

; Inputs:

; none

; Outputs:

; r0 <- 1 if button pressed (0 otherwise)

; r1 <- button number pressed (0 to 15)

GetKeyPress:

push r16 ; save registers

push r18

push r19

push r20

push r22

clr r0

clr r1

ldi r16, 0b11101111 ; clear output pins PA4-PA7

clr r20 ; initial button number

KeyLoop:

out PORTA, r16

; short delay while changes to row voltages propagate through keypad

ldi r22, 100

Delay:

dec r22

brne Delay

; read status of buttons from keypad

in r18, PINA ; read values from PA0-PA3

com r18

andi r18, 0x0F ; check if any buttons pressed

breq RowNotLit

; if a button was pressed, compute the button number (0 to 15)

sbrc r18, 1 ; if 2nd button in row pressed

subi r20, -1 ; then add 1 to button number for 2nd button in row

sbrc r18, 2 ; if 3rd button in row pressed

subi r20, -2 ; then add 2 to button number for 3rd button in row

sbrc r18, 3 ; if 4th button in row pressed

subi r20, -3 ; then add 3 to button number for 4th button in row

ldi r19, 0x01

mov r0, r19 ; indicate a button was pressed (set r0 to 1)

mov r1, r20 ; copy button number pressed into r1

rjmp ExitProc

RowNotLit:

; if button not pressed, check next row in keypad

lsl r16

ori r16, 1

subi r20, -4 ; update button number for next row

; if all rows checked, exit procedure

cpi r16, 0b11111111

brne KeyLoop

ExitProc:

pop r22 ; restore registers

pop r20

pop r19

pop r18

pop r16

ret

MyDelay:

push r18

push r19

Ldi r18, 2

Ldi r19, 0x05

Out TCCR0, r19; Timer starts counting from 0 at Microcontroller clock rate/1024

Timer1:

Ldi r19, 6

Out TCNT0, R19 ; Corrects from 512ms to 500ms

Timer:

IN r19, TIFR

SBRS R19, 0; Checks the Top Overflow flag (TOV0) in the TIFR

Rjmp timer; loops until the flag gets set

Ldi r19, 0x01

Out TIFR, R19 ; Write a '1' over the TOV0 flag to clear it

Dec r18;

Brne timer ; Controls the number of loops for the delay. Using 2 to reach a .5s delay

clr r19

Out TCCR0, r19; Stop the timer

pop r19

pop r18

ret

InterRout0:

cpi r21, 10

brsh EndInter0 ; check if max delay

inc r21; if not, increase delay

EndInter0: reti

InterRout1:

cpi r21, 2

brlo EndInter1 ; check if max delay

dec r21; if not, increase delay

EndInter1: reti

TextArray:

.DB "11112abc3def", 0, 0, 0, 0

.DB "4ghi5jkl6mno", 0, 0, 0, 0

.DB "7prs8tuv9wxy", 0, 0, 0, 0

.DB " ,.\*0000####", 0, 0, 0, 0

.DSEG

TextString: .byte 1