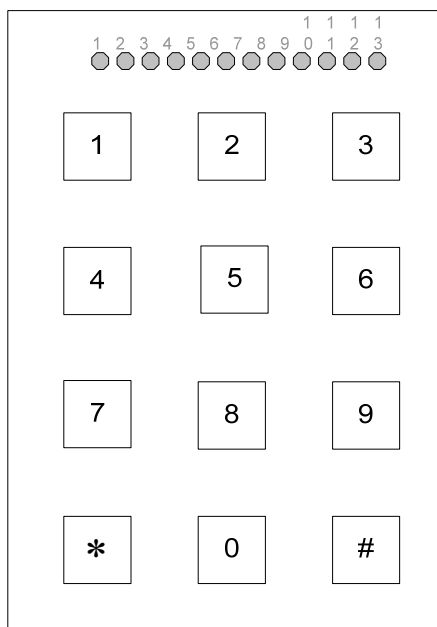


Front View



Key	1	2	3	4	5	6	7	8	9	*	0	#	
Pin	5	9	13	4	8	12	3	7	11	2	6	10	1

	7	6	5	4	3	2	1	0
reset :	PUPKE	0	0	PUPEE	0	0	PUPBE	PUPAE
	1	0	0	1	0	0	0	0

PUPKE: pull-up Port K enable

0 = Port K pull-up resistors are disabled

1 = Port K pull-up resistors are enabled

PUPEE: pull-up Port E enable

0 = Port E input pins 7 and 4-0, pull-up resistors are disabled

1 = Port E input pins 7 and 4-0, pull-up resistors are enabled

PUPBE: pull-up Port B enable

0 = Port B pull-up resistors are disabled

1 = Port B pull-up resistors are enabled

PUPAE: pull-up Port A enable

0 = Port A pull-up resistors are disabled

1 = Port A pull-up resistors are enabled

Figure 7.8 Pull-Up control register

```

; PORTA connected to keypad

; PA0 connected to pin 2 (*)
; PA1 connected to pin 3 (7)
; PA2 connected to pin 4 (4)
; PA3 connected to pin 5 (1)
; PA4 connected to pin 6 (0)
; PA5 connected to pin 7 (8)
; PA6 connected to pin 8 (5)
; PA7 connected to pin 9 (2)

; PORTB connected to keypad
; PB0 connected to pin 10 (#)
; PB1 connected to pin 11 (9)
; PB2 connected to pin 12 (6)
; PB3 connected to pin 13 (3)
; PB4 connected to pin 1

```

Note: PB4 is set as output

```
#include "9s12.inc"
```

```

;mainline of code
    org    $2000
    lds    # $2000           ; define stack address
    movb   # $03, PUCR      ; enable pull up resistors on PORTs A & B
    movb   # $00, DDRA      ; PORT A configured for input
    movb   # $F0, DDRB      ; bits 7~4 for output, bits 3~0 for input
    jsr    getchar          ; call subroutine
    swi

getchar
scan_k0   brset   PORTA, $01, scan_k1   ; is key * pressed?
          jsr     delay10ms             ; debounce key *
          brset   PORTA, $01, scan_k1
          ldaa    # $2A                  ; get the ASCII code of *
          rts

scan_k1   brset   PORTA, $02, scan_k2   ; is key 7 pressed?
          jsr     delay10ms             ; debounce key 7
          brset   PORTA, $02, scan_k2
          ldaa    # $37                  ; get the ASCII code of 7
          rts

scan_k2   brset   PORTA, $04, scan_k3   ; is key 4 pressed?
          jsr     delay10ms             ; debounce key 4
          brset   PORTA, $04, scan_k3
          ldaa    # $34                  ; get the ASCII code of 4
          rts

```

```

scan_k3  brset  PORTA,$08,scan_k4    ; is key 1 pressed?
        jsr    delay10ms           ; debounce key 1
        brset  PORTA,$08,scan_k4
        ldaa   #$31                ; get the ASCII code of 1
        rts

scan_k4  brset  PORTA,$10,scan_k5    ; is key 0 pressed?
        jsr    delay10ms           ; debounce key 0
        brset  PORTA,$10,scan_k5
        ldaa   #$30                ; get the ASCII code of 0
        rts

scan_k5  brset  PORTA,$20,scan_k6    ; is key 8 pressed?
        jsr    delay10ms           ; debounce key 8
        brset  PORTA,$20,scan_k6
        ldaa   #$38                ; get the ASCII code of 8
        rts

scan_k6  brset  PORTA,$40,scan_k7    ; is key 5 pressed?
        jsr    delay10ms           ; debounce key 5
        brset  PORTA,$40,scan_k7
        ldaa   #$35                ; get the ASCII code of 5
        rts

scan_k7  brset  PORTA,$80,scan_k8    ; is key 2 pressed?
        jsr    delay10ms           ; debounce key 2
        brset  PORTA,$80,scan_k8
        ldaa   #$32                ; get the ASCII code of 2
        rts

scan_k8  brset  PORTB,$01,scan_k9    ; is key # pressed?
        jsr    delay10ms           ; debounce key #
        brset  PORTB,$01,scan_k9
        ldaa   #$23                ; get the ASCII code of #
        rts

scan_k9  brset  PORTB,$02,scan_k10   ; is key 9 pressed?
        jsr    delay10ms           ; debounce key 9
        brset  PORTB,$02,scan_k10
        ldaa   #$39                ; get the ASCII code of 9
        rts

scan_k10 brset  PORTB,$04,scan_k11   ; is key 6 pressed?
        jsr    delay10ms           ; debounce key 6
        brset  PORTB,$04,scan_k11
        ldaa   #$36                ; get the ASCII code of 6
        rts

scan_k11 brset  PORTB,$08,keyloop    ; is key 3 pressed?
        jsr    delay10ms           ; debounce key 3
        brset  PORTB,$08,keyloop
        ldaa   #$33                ; get the ASCII code of 3
keyloop  jmp    scan_k0

```

```
    rts
```

```
;the following subroutine creates a delay of 10 ms
```

```
delay10ms  movb    #$90,TSCR1           ; enable TCNT and fast flags clear
           movb    #$06,TSCR2         ; set prescale factor to 64
           movb    #$01,TIOS          ; enable OC0
           ldd     TCNT
           add     #$3750              ; start an output compare operation
           std     TCO                 ; with 10 ms time delay
wait       brclr   TFLG1,$01,wait     ; if equal, COF in TFLG1 is set to 1
           rts
```