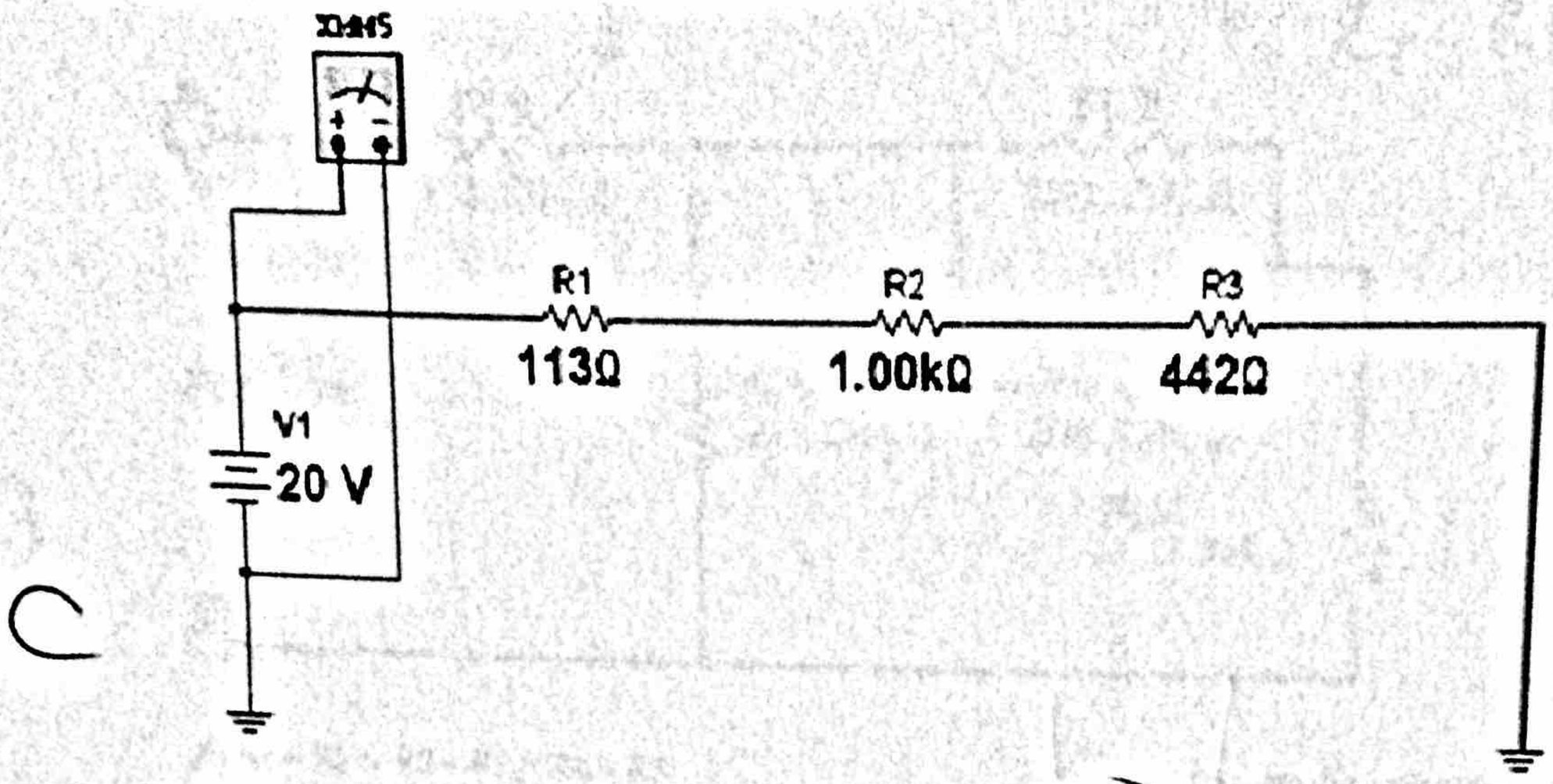
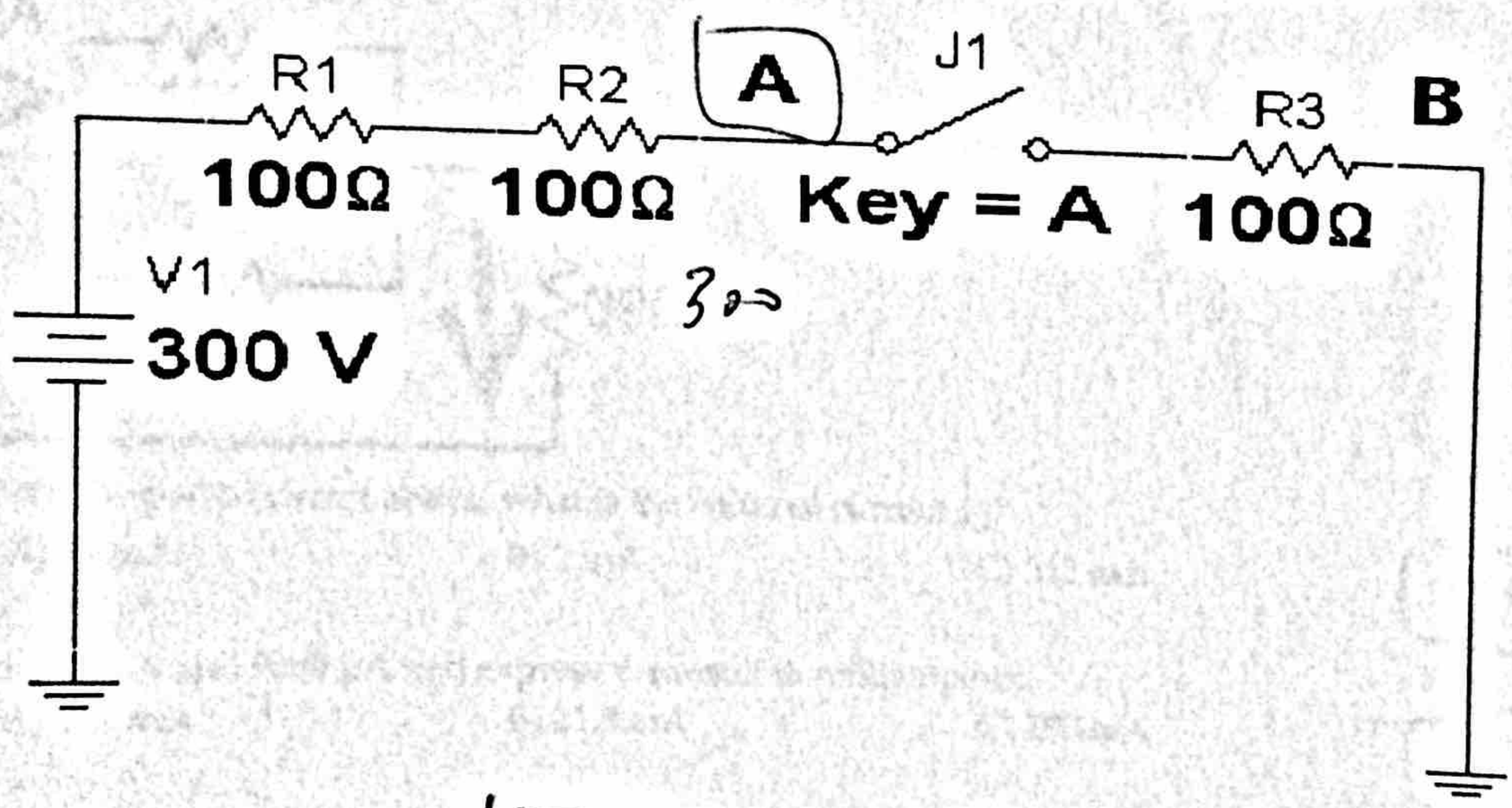
12) What is being measured in the following circuit?



C

A) Rt B) It C) V1 D) VR1

13) With the switch in the position shown, what is the value of V_A in the following circuit?



B

A) 0 V B) 300 V C) 100 V D) 200 V

14) The opposition to the flow of current is called _____.

A) capacitance B) current C) resistance D) voltage

C

0101 1100 1010 0011 1010 0100 65 536 4096 0110 1001 1010

Number Conversions – (5 marks) Perform the following number conversions, placing your answers in the **Value** column:

Convert	To	Value
$F8 (unsigned)	Integer	248
$F8 (signed)	Integer	-8
$1FFF (unsigned)	Integer	61440
92 0101 1100	8-bit Hexadecimal (signed 2's complement	$5C
-92	8-bit Hexadecimal (signed 2's complement	$53
-6	8-bit Hexadecimal (signed 2's complement)	$AA
128	Binary	1000 0000
-128	Binary (signed 2's complement)	1000 0000
1111 1110 (signed)	Integer	-2
1111 1110 (unsigned)	Integer	254

-2

0000 0001 255 254 0000 0010 100 8 256 0-15 16=16 8=256 0110 1001 1010 F8 1111 1000

BCD Arithmetic Methods – (5 marks) To confirm your understanding of BCD Arithmetic methods, represent 265_{10} and 135_{10} in BCD and then correctly perform BCD addition. **Show all steps** (including all annotations to the right of the additions) for full credit.

265_{10} + 135_{10} = 400_{10}

0010 0110 0101 ← 265 in BCD

+ 0001 0011 0101 ← 135 in BCD

0011 1001 1010 ← INVALID BCD LOWER NIBBLE "A" or 10 (INVAL.) INC 6

+ 0000 0000 0110 + 0006 For correcting lower nibble

0011 1010 0000 + INVALID BCD - CARRY OVER FROM LOWER NIBBLE (CAUSE NIBBLE = 'A')

+ 0060 0110 0000 + 060 For correcting Both nibble

0100 0000 0000 ← BCD for 400

4 0 0

1000 0010
0000 0000
0111 1111
1000 0001

-2

F8
1111 1000
0000 0111
1 08

Program Tracing (10 marks)

Given the following program listing, trace the results (hexadecimal value of registers being asked) of each program step **after** the line of code has been executed, as per the example answer for SP. *(Instruction Set included with this Term Test).*

```
1  ; Trace1_VB.asm
2  ; Name            David Haley
3  ; S/N             Faculty
4
5
6  ; Purpose         To manually trace through given Assembly Language code
7
8  VALUE1   equ      $5678
9  VALUE2   equ      $21
10
11          org      $1000
12 Source   db       %00111100, $82, $54, 128
13 Source2  dw       $1234
14
15
16                                ; All Values must be in Hexadecimal as per the
17          org      $2000        ; example for lds
18          lds      #$2000       ; S = $2000
19
20          ldd      #Value1      ; A = $56
21
22                                ; B = $78
23
24          aba                   ; A = $CE
25
26                                ; B = $78
27
28          ldaa     Source+3     ; A = $80
29
30          ldab     $1000        ; B = $3C
31
32          ldx      Source2      ; X = $1234
33
34          ldx      #Source2     ; X = $1004
35
36          ldaa     #$02
37          ldab     #$01
38          exg      a,b          ; A = $01
39
40          ldy      Source+4     ; Y = $1234
41
42
43          end
```

Programming Concepts – Analysis of Code (10 marks)

Given the following source code listing, indicate the LEDs' output pattern for two complete iterations of the for (;;) [forever] loop by completing the table that follows the code listing. **Note that this is fully functional code, which correctly assembles and outputs a specific pattern on Port B's LEDs.**

To indicate that a specific LED is ON. place an X in the appropriate box in the table as per the following example (*which you should note is NOT part of the pattern produced by the code listing*). If a specific LED is OFF, leave the appropriate box in the table blank.

Time (msec)	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
3000	X			X			X	

```
;Walk_A_Bit_Structed_V2_Short_Version.asm
#include C:\68hcs12\registers.inc
; Program Constants
STACK   equ     $2000
; Delay_MS Constants
DVALUE  equ     $250    ; Delay value (base 10) 0 - 255 ms

        org     $1000   ; data area
LEDTABLE                ; values to display on LEDs
        db      $81, $42, $24, $18, $00
ENDTABLE

        org     $2000   ; program code
Start   lds     #STACK  ; stack location
        jsr     Config_SWs_and_LEDs

; Continually Flash LEDs with values from LEDTABLE, changing values every 250 ms
; for (;;)
Back    ldx     #LEDTABLE       ; point to first element of LEDTABLE
Again   ldaa    1,x+            ; get first value, increment pointer
        staa    portb           ; output value to LEDs
        ldaa    #DVALUE         ; delay value
        jsr     Delay_ms        ; delay routine
        cpx     #ENDTABLE       ; one less to do
        bne     Again           ; done yet?
        bra     Back            ; endless loop

; Predefined Subroutines Follow
#include C:\68HCS12\LIB\Config_SWs_and_LEDs.asm
#include C:\68HCS12\LIB\Delay_ms.asm
        end
```

Expanded view of LEDTABLE
$81, $42, $24, $18, $00

Indicate the LEDs' output pattern by completing this table (1 mark per correct line):

Time (msec)	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
0	X							X
250		X					X	
500			X			X		
750				X	X			
1000								X
1250	X						X	
1500		X				X		
1750			X					
2000				X	X			
2250								

← no LEDs

← no LEDs

Program Tracing (10 marks)

Given the following program listing, trace the results (hexadecimal value of registers being asked) of each program step **after** the line of code has been executed, as per the example answer for SP. *(Instruction Set included with this Term Test).*

```
1  ; Trace2_VB.asm
2
3  ; Name                David Haley
4  ; S/N                 Faculty
5
6
7  ; Purpose             To manually trace through given Assembly Language code
8            org         $1000
9  Data1     db          $02, $33, $FF, $06, $81
10 Data2     db          $04, $99, $9F, $40, $FE, $01, $23, $94, $01
11
12
13                       ; All Values must be in Hexadecimal as per the
14           org         $2000                    ; example for lds
15           lds         #$2000                   ; SP =?   S2000
16
17           ldy         #Data1                   ; Y = ?   [illegible]  $1000
18
19           ldaa        1,y+                     ; A = ?   $02
20
21                                                ; Y = ?   $1001
22
23           ldab        0,y                      ; B = ?   $33
24
25                                                ; Y = ?   $1001
26
27           ldx         #Data2                   ; X = ?   $1005
28
29           ldaa        4,x                      ; A = ?   $FE
30
31           inx         X = $1006
32
33           ldd         4,x+                     ; A = ?   $99
34
35                                                ; B = ?   $9F
36
37                                                ; X = ?   [illegible]  $100A
38
39           swi
40           end
```