(5) **Question 1.** A) Finish instruction, push registers, I=1, PC=vector, execute ISR.

(5) **Question 2.** \(2^{12}=4096\), \(5V/4096\), which is about 1.2mV

(5) **Question 3.** (We can reduce dropout by multiplying before divide by 100)
Start with the goal of the problem: \(A = H*W\)
Specify the fixed-point definitions: \(A = IA/100, W = IW/100, H = IH/100\)
Substitute definitions into problem: \(IA/100 = IW/100 * IH/100\)
Solve algebraic \(IA = (IW * IH)/100\)

**Questions 4 and 5** Hand execute and build a stack picture

```
 movw #1000,2,-sp  SP → In 0
   In 1000
     jsr sub2  In 1000
     return

 SP → Cnt
     return

 psha pshx
 SP → 0,SP  1,SP  2,SP  3,SP  4,SP  5,SP  6,SP  7,SP
 OldX
 OldA
 Cnt
```

(5) **Question 4.** \texttt{in} set 6 ; binding of 16-bit input parameter

(5) **Question 5.** \texttt{cnt} set 3 ; binding of 8-bit local variable

(15) **Question 6.** Any resistor set that is a power of 2 is OK.

![Resistor Diagram](resistor_diagram.png)

(5) **Question 7.** C) The software writes to the ATDCTL5 register.

(5) **Question 8.** \texttt{fdb Brake,Go}

(5) **Question 9.** \texttt{ldab 0,y} (or \texttt{ldab 1,y+})

(5) **Question 10.** \texttt{ldy 1,y} (or \texttt{ldy 0,y})

(5) **Question 11.** C) Because the \texttt{Fifo} queue decouples execution of producer and consumer.

(5) **Question 12.** D) All of A, B, and C are correct.
(30) Question 13. A system that increments an 8-bit variable called Second, every 1 second.

```assembly
org $3800 ; RAM
Second rmb 1 ; increment this every second

org $4000 ; EEPROM
main lds #$4000 ; initialize stack
clr Second ; initialize shared global
movb #$80,TSCR1 ; enable TCNT
movb #$07,TSCR2 ; divide by 128
bset TIOS,#$80 ; enable OC7
bset TIE,#$80 ; ARM OC7
ldd TCNT
addd #31250
std TC7 ; first one in 1 second
asm cli ; enable
loop bra loop ; main program does nothing

; output compare 7 interrupt service routine, every 1 sec
; 4,000,000 E clocks per second
; 4,000,000/128 = 31250 TCNTs per second
OC7han
movb #$80,TFLG1 ; acknowledge
ldd TC7
addd #31250
std TC7 ; next interrupt in 1 second
inc Second
rti

org $FFE0
fdb OC7han ; output compare 7 interrupt vector

org $FFFE
fdb main ; reset vector

; 4000000/2 = 2000000
; 4000000/4 = 1000000
; 4000000/8 = 500000
; 4000000/16 = 250000
; 4000000/32 = 125000
; 4000000/64 = 62500
; 4000000/128 = 31250
```