(4) Question 1. We can hear sounds from about 20 to 20 kHz, thus we say the frequency components of sound exist from 20 to 20 kHz. The Nyquist Theorem requires you to sample at a rate strictly larger than twice the maximum frequency components of the signal. Thus in this case, we must sample larger than 800 Hz. Recall in lab the distance signal was 0 to 2 Hz, and we sampled at 5 Hz.

(4) Question 2. A) The motor will not spin at all because we did not wait in between each output to the motor.

(4) Question 3. Write a subroutine to sample ADC channel 4 of the 9S12DP512.

```
ADC_In movb #$84,ATD0CTL5           ;start ADC channel 4
       brclr ATD0STAT0,#$80,adcwait ; wait for SCF
       ldx   ATD0DR0
       rts
```

(8) Question 4. 0 maps to 0 and 2047 maps to 1000. When Vin is 10 V, the ADC is 2047, software converts 2047 into 1000. Multiply by 1000, then divide by 2047

```
   ldy  #1000
   emul
   ldx  #2047
   ediv
```

(4) Question 5. The trick is the `sty` instruction overwrites the data saved on the stack by the `stx` instruction. RegX will be $7855

(4) Question 6. The order is B) 1,6,3,2,5,4,7

1) There is data in the receive data register and the hardware sets the flag bit (e.g., RDRF=1)
6) The CCR, A, B, X, Y, PC are pushed on the stack
3) The I bit in the CCR is set by hardware
2) The SCI vector address is loaded into the PC
5) The software reads `SCI1SR1`
4) The software reads `SCI1DRL`
7) The software executes `rti`

(10) Question 7. Write software that increments a 16-bit global variable every 1 msec

```
       org  $0800
Count  rmb  2  ;incremented every 1 msec
       org  $4000
main   lds  #$4000
       movw #0,Count
       movb #$80,TSCR1  ;enable TCNT
       bset TIOS,#$04   ;activate output compare
       bset TIE,#$04    ;arm OC2
       ldd  TCNT
       addd #50
       std  TC2         ;first interrupt right away
       cli
       bra  *
OC2han movb #$04,TFLG1  ;ack
       ldd  TC2
       addd #8000       ;1ms=8000*125ns
       std  TC2         ;next interrupt
       ldx  Count
```

```
        inx
        stx  Count
        rti
        org  $FFEA
        fdb  OC2han
        org  $FFFE
        fdb  main
```

(5) Question 8. Idle is high, start is low, bits go 0,1,2,3,4,5,6,7, stop is high, idle is high

```
      | ---- |      |      |      | ---- |      | ---- |      | ---- |      | ----- | ----- | ----- | ----
  PS3 |      |      |      |      |      |      |      |      |      |      |       |       |       |
      |      | ---- | ---- | ---- |      | ---- |      | ---- |      | ---- |       |       |       |
  t  -1      0     100    200    300    400    500    600    700    800    900    1000    1100    1200    us
```

(5) Question 9. Transmits 1 to 2, then 2 to 1 taking a total of 20 ms.

(4) Question 10. The most common mistake was to think V_{out} is 5V when the digital output was 111. Let $n=4b_2+2b_1+b_0$ be the 3-bit number on PT2,PT1,PT0. The voltage on PT2 is $5*b_2$. The voltage on PT1 is $5*b_1$. The voltage on PT0 is $5*b_0$. We will define all currents going from left to right. The current through the 10kΩ is $(5*b_2-V_{out})/10k\Omega$. The current through the 20kΩ is $(5*b_1-V_{out})/20k\Omega$. The current through the left 40kΩ is $(5*b_0-V_{out})/40k\Omega$. The current through the right 40kΩ is $V_{out}/40k\Omega$. Adding the currents up we get

$$(5*b_2-V_{out})/10k\Omega + (5*b_1-V_{out})/20k\Omega + (5*b_0-V_{out})/40k\Omega = V_{out}/40k\Omega.$$

Multiply by 40kΩ

$$4(5*b_2-V_{out}) + 2(5*b_1-V_{out}) + (5*b_0-V_{out}) = V_{out}$$

Solving for V_{out}

$$4*5*b_2 + 2*5*b_1 + 5*b_0 = 8*V_{out}$$

Substituting the definition of n

$$V_{out} = 5*n/8$$

PT2	PT1	PT0	Vout (V)
0	0	0	0.000
0	0	1	0.625
0	1	0	1.250
0	1	1	1.875
1	0	0	2.500
1	0	1	3.125
1	1	0	3.750
1	1	1	4.375

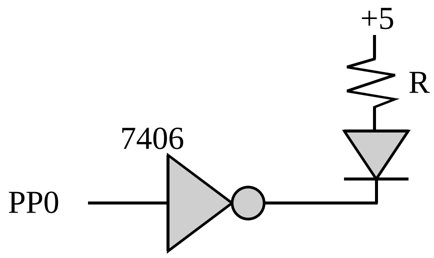
(8) Question 11. You will be execute one instruction and answering questions.

Part a) 07 is the machine code for **`bsr`**

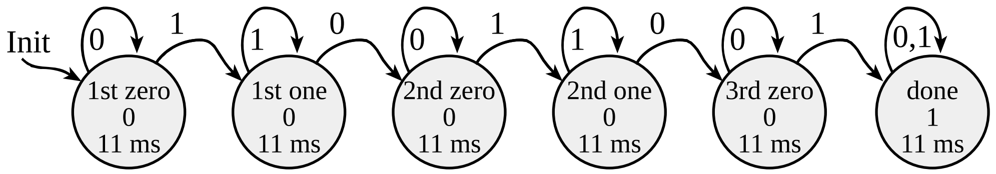
Part b) The return address is $4005, which is pushed on the stack. To push, first decrement SP then store. The order in this table does not matter. In fact, the 9S12 will store the $4005 as a 16-bit value in one bus cycle

Address	Data
$3FFD	$05
$3FFC	$40

Part c) The SP points to valid data on top of the stack, SP = $3FFC
Part d) The PC is incremented after each fetch (PC=$4005), then the PC relative value of $02 is added to the PC to get the target address of the subroutine, PC = $4007
(5) Question 12. R = $(5\text{-}V_d\text{-}V_{OL})/I_d$ = (5-2-0.5)/0.01 = 2.5V/0.01A = 250 Ω.



(10) Question 13. Basically, we wait for the input to be 1, wait for it to be zero, wait for the input to be 1, wait for it to be zero, then wait for the input to be 1. To debounce we run the FSM at any time between 10 and 100 ms.



(10) Question 14.
Part a) Hand execute and draw a stack picture

```
Pt set  5  ;16-bit pointer to 8-bit data
L1 set  2  ;8-bit local variable
```

Part b) A pointer is on the stack, so we need an indirection to get the data

```
 ldaa [Pt,y]
```

We could have performed the indirection in multiple instructions

```
 ldx  Pt,y
 ldaa 0,x
```

(15) Question 15. The FIFO is full when Cnt is 8.

```
Fifo_Put ldaa Cnt    ;number of elements currently saved, 0 to 8
         cmpa #8
         bhs  full   ;full if Cnt is 8
NotFull  ldx  #Fifo
         ldaa PutI   ;RegA is 0 to 7
         stab a,x    ;save in Fifo
         inc  Cnt    ;one more saved
         inca        ;next place to put
         anda #$07   ;0 to 7
         staa PutI
         ldaa #1     ;success
         bra  done
full     clra        ;failure
done     rts
```