EE302 sem 1 exam 2019-2020

Q1 (c) (i) I/O range is 0x1C00 .. 0x1CFF = 0xFF ⇒ 0xFF + 1 addresses
= 256 addresses

256 ✓

(ii)

RAM {

RAM	0x0 — 0x0FFF
unused RAM	0x1000 — 0x17FF
EEPROM	0x1800 — 0x1BFF
UART	0x1C00 — 0x1C03
/ / /	0x1C04 — 0x1C05
Inertial	0x1C06 — 0x1C0F
/ / / /	0x1C10 — 0x1CFF
unmapped	0x1D00 — 0x1DFF
Bootloader	0x1E00 — 0x1FFF

extend method

(subrange 1 2 start at 0x1C06
" 8 " " 0x1C08)

(iii) Bootloader 0x1E00 = 1 1110 0000 0000 (Basic method)
0x1FFF = 1 1111 1111 1111

$\overline{BL\_ROM\_CS} = \overline{(A_{12} A_{11} A_{10} A_9)}$ $\sim(A_{12} A_{11} A_{10} A_9)$

EEPROM 0x1800 = 1 1000 0000 0000 (basic method)
0x1BFF = 1 1011 1111 1111

$\overline{EEPROM\_CS} = \overline{(A_{12} A_{11} \bar{A}_{10})}$ $\sim(A_{12} A_{11} \sim A_{10})$

Inertial Sensor : 10 registers not a power of 2 ⇒ Extended Method

subrange 0x1C06 1 1100 0000 0110
0x1C07 1 1100 0000 0111

分 2 段

subrange1 = $A_{12} A_{11} A_{10} \bar{A}_9 \bar{A}_8 \bar{A}_7 \bar{A}_6 \bar{A}_5 \bar{A}_4 \bar{A}_3 A_2 A_1$

subrange2 : 0x1C08 1 1100 0000 1000
0x1C0F 1 1100 0000 1111

= $A_{12} A_{11} A_{10} \bar{A}_9 \bar{A}_8 \bar{A}_7 \bar{A}_6 \bar{A}_5 \bar{A}_4 A_3$

$\overline{Inertial\_CS} = \overline{(subrange1 + subrange2)}$

$\sim(S1 + S2)$

Assumption:
(Active low
chip selects)

Q1 (b) i) 19200 — baud rate

8 — # data bits per async word

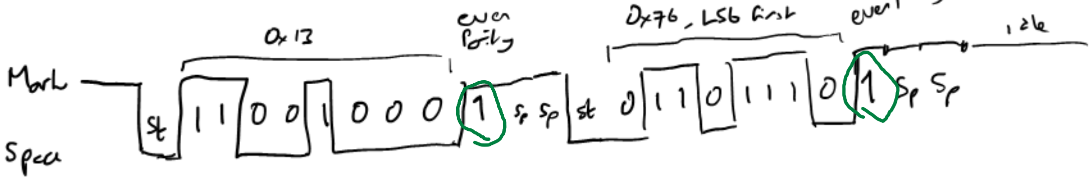
E — even parity

2 — # stop bits per async word

ii) assume order of words transmitted is MS Word first, then least significant word.

Data is <u>0x1376</u> ⇒ 2 words <u>0x13, 0x76</u>   13   76

⇒ binary <u>0001 0011</u>, <u>0111 0110</u>



iii) Max rate for sample values

Each sample is 2 bytes data ⇒ 2 async words

Each async word is 12 bits long

baud rate was 19200 — $T_{bit} = \frac{1}{19200}$

$\Rightarrow T_{word} = \frac{12 \times 1}{19200} = 625 \mu s$

2 words $\Rightarrow T_{sample} = 2 \times 625 \mu s = 1.25 ms$

$F_{sample} = \frac{1}{T_{sample}} = \underline{\underline{800 \text{ samples/sec}}}$

Q2 (a) theory - see notes

(b) (i) bitmask 0b0001 1110 = 0x1E in hex

(ii) Mask = 0x1E

oldscale = (VCON & Mask) >> 1

```
// VCON          0xEB      0b 1110 1011
// mask          0x1E      0b 0001 1110
                           ------------
// VCON & mask             0b 0000 1010
//       >> 1              0b 0000 0101 = 0x05
// => oldscale is 0x05
```

(iii) $1.375V = \text{Scale} \times V_{DD}/24 = \text{scale} \times \frac{3.3}{24}$

$\Rightarrow \text{Scale} = \frac{1.375 \times 24}{3.3} = 10$

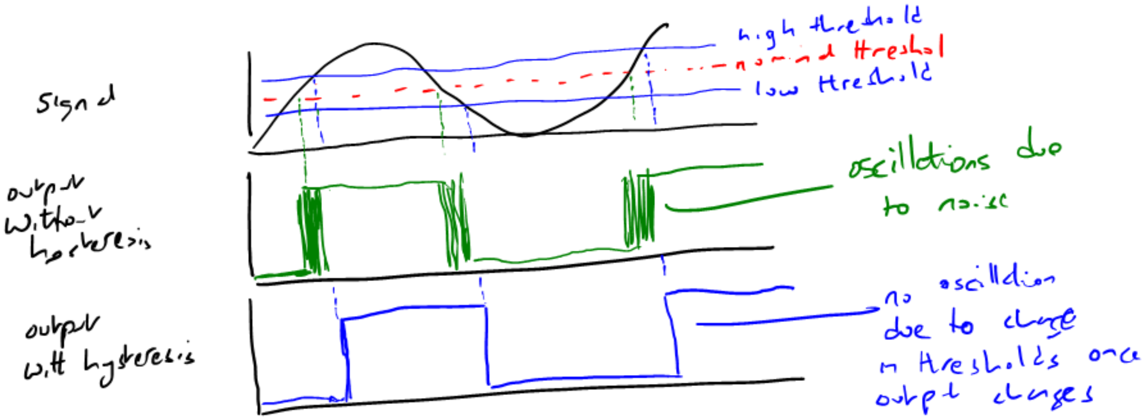
$\Rightarrow$ newscale = 10 = 0xA = 0b0000 1010

$$\text{VCON} = \underbrace{(\text{VCON} \,\&\, \sim\text{Mask})}_{\text{tmp1}} \mid \underbrace{(\text{newScale} \ll 1)}_{\text{tmp2}}$$

```
// VCON = 0xEB             0b1110 1011
// ~Mask                   0b1110 0001
                           -----------
// tmp1                    0b 1110 0001
// newscale   0xA          0b 0000 1010
//    << 1                 0b 0001 0100
//   tmp2                  0b 0001 0100
// tmp1 | tmp2 = VCON      0b 1111 0101
                           = 0x F5
```

tmp1 | tmp2
bitwise OR

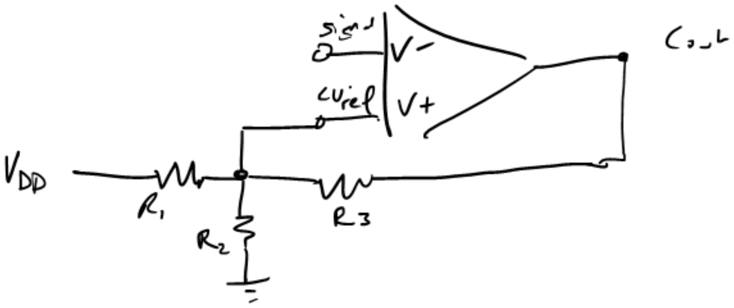
Q2 (c) (i)



Signal
high threshold
nominal threshold
low threshold
output without hysteresis
oscillations due to noise
output with hysteresis
no oscillation due to change in thresholds once output changes

More explanation needed - see notes

Q2 (c) (ii)



R_1 and R_2 form a basic resistor divider

R_3 acts in parallel with R_1 if Cout is high or R_2 if Cout is low.

For additional detail required, see notes

(iii) $V_{DD} = 3.3V$, $V_{SS} = 0V$, nominal threshold is 1.1V

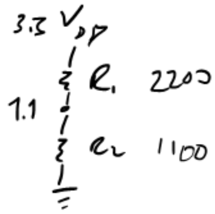
Noise is $< \pm 0.2V$ $\Rightarrow$ $V_{LOW} = 1.1 - 0.2 = 0.9V$

$V_{HIGH} = 1.1 + 0.2 = 1.3V$

Ref circuit should not use much current (eg 1mA),

so choose $R_1 + R_2 \simeq 3300\Omega$

$\Rightarrow R_1 \simeq 2200\Omega$



From notes

$$\frac{R_2}{R_1} = \frac{V_{LOW}}{V_{DD} - V_{HIGH}} = \frac{0.9}{(3.3 - 1.3)} = 0.45 \Rightarrow R_2 = 0.45 \times 2200 = 990\Omega$$

$$\frac{R_3}{R_1} = \frac{V_{LOW}}{V_{HIGH} - V_{LOW}} = \frac{0.9}{1.3 - 0.9} = 2.25 \Rightarrow R_3 = 2.25 \times 2200 = 4950\Omega$$

Q3 (a)

```
updateLED():
    constant FLASH_DURATION_TICKS = 6
    static ledOnCounter = 0

    if gFlashNeeded is TRUE
        set ledOnCounter = FLASH_DURATION_TICKS

    if ledOnCounter > 0
        set LED to HIGH    // on
        decrement ledOnCounter
    else
        set LED to LOW    // off
```

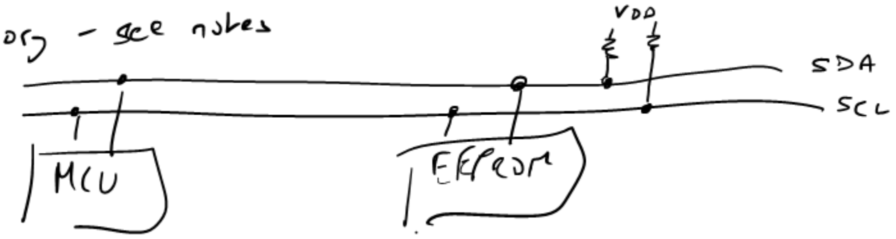
(b) (i) Theory – see notes

(ii) When the ISR runs

- it first checks ADCIF and finds ADC interrupt is present so it would handleAdcInterrupt and clear the flag to show we've finished with this interrupt. At this point the ISR returns.
- since there is still an active interrupt (the USART transmit) the ISR would be called again. We would check ADCIF (not active) PORTIF (not active) and finally TXIF which is active. We would handleUsartTxInterrupt and clear it and ISR returns

(c) (i) Theory – see notes

(ii)



(iii) Assuming data at location 0x470 and 0x471

// set the starting location

START I2C Addr $\overline{W}$ ACK

0x4 ACK

0x70 ACK

stop

—— = controller

no red line = peripheral

// read the data

Start I2C Addr R ACK

0x81 ACK

0x82 NACK

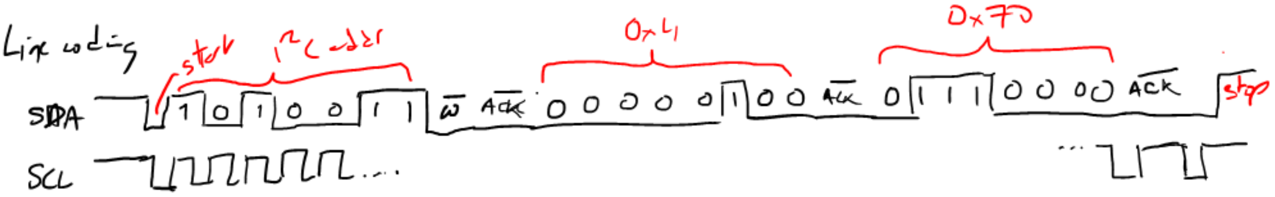
Stop

Q3 (c) (iv)

I2C addr : 0x53 = 0b 101 0011 (7 bit addr)

0x470 = 0b 0000 0100 0b 0111 0000 (2 bytes data)

Line coding



Q4 (a) Mainly theory – see notes

(b) (i) theory – see notes

(ii) $f_{osc} = 10\,MHz$, $f_{src} = \frac{f_{osc}}{4} \Rightarrow T_{src} = \frac{4}{10\,MHz} = 0.4\mu s$

1:1 Timeout $T_{src} \times pre \times 2^{16} \times post$

$0.4\mu s \times 1 \times 2^{16} \times 1 = 26.2144\ ms$

Resolution $T_{src} \times pre = 0.4\mu s$

1:8 Timeout $0.4 \times 8 \times 2^{16} \times 1 = 209.7152\ ms$

resolution $0.4 \times 8 = 3.2\mu s$

(iii) $F = 18\ Hz$ rate $\Rightarrow T = \frac{1}{18} = 55.55\dot{5}\ ms$

Allow $3\mu s$ for interrupt and reset $\Rightarrow T_{intr} \approx 55.552\ ms$

desired prescale $= \frac{T_{intr}}{(T_{src} \times 2^{16})} = \frac{55\,552\ \mu s}{0.4\mu s \times 65536} = 2.119$

$\Rightarrow$ required prescale is 1:4

$N_{intr} = \text{round}\left(\frac{T_{intr}}{T_{src} \times pre}\right) = \text{round}\left(\frac{55552}{0.4 \times 4}\right) = 34720$

$\Rightarrow$ Timer1 reg must be set to $2^{16} - 34720 = 30816$

(iv) Timeout will occur in $3 + (34720 \times 0.4 \times 4) = 55555\,\mu s$

Desired period was really $55555.5\dot{5}\,\mu s$

Percentage error is $100 \times \left(\frac{55555.5\dot{5} - 55555}{55555.5\dot{5}}\right) \simeq 0.00099\,\%$

Precise timing is not possible due to the limited resolution of the clock after prescaling. ($1.6\mu s > 0.5\dot{5}\mu s$ we need).