Q1)

Startup.s

Modified part is:

```
Timer0A_Handler
                IMPORT  My_Timer0A_Handler
TIMER0A_Handler PROC
                EXPORT  TIMER0A_Handler  [WEAK]
                ;POP    {R0}
                PUSH    {LR}
                BL      My_Timer0A_Handler
                POP     {LR}
                ;PUSH   {R0}
                BX      LR
                ENDP
```

Pulse.s

```
; Pulse.s
; Routine for creating a pulse train using interrupts
; This uses Channel 0, and a 1MHz Timer Clock (_TAPR = 15 )
; Uses Timer0A to create pulse train on PF2

;Nested Vector Interrupt Controller registers
NVIC_EN0_INT19          EQU 0x00080000 ; Interrupt 19 enable
NVIC_EN0                EQU 0xE000E100 ; IRQ 0 to 31 Set Enable Register
NVIC_PRI4               EQU 0xE000E410 ; IRQ 16 to 19 Priority Register

; 16/32 Timer Registers
TIMER0_CFG              EQU 0x40030000
TIMER0_TAMR             EQU 0x40030004
TIMER0_CTL              EQU 0x4003000C
TIMER0_IMR              EQU 0x40030018
TIMER0_RIS              EQU 0x4003001C ; Timer Interrupt Status
TIMER0_ICR              EQU 0x40030024 ; Timer Interrupt Clear
TIMER0_TAILR        EQU 0x40030028 ; Timer interval
TIMER0_TAPR             EQU 0x40030038
TIMER0_TAR              EQU     0x40030048 ; Timer register

;GPIO Registers
GPIO_PORTF_DATA         EQU 0x40025010 ; Access BIT2
GPIO_PORTF_DIR          EQU 0x40025400 ; Port Direction
GPIO_PORTF_AFSEL    EQU 0x40025420 ; Alt Function enable
GPIO_PORTF_DEN          EQU 0x4002551C ; Digital Enable
GPIO_PORTF_AMSEL    EQU 0x40025528 ; Analog enable
GPIO_PORTF_PCTL     EQU 0x4002552C ; Alternate Functions

;System Registers
SYSCTL_RCGCGPIO     EQU 0x400FE608 ; GPIO Gate Control
SYSCTL_RCGCTIMER    EQU 0x400FE604 ; GPTM Gate Control
```

```
;-------------------------------------------------
LOW                             EQU     0x00000300
HIGH                            EQU     0x00000300
;-------------------------------------------------

                        AREA    routines, CODE, READONLY
                        THUMB
                        EXPORT          My_Timer0A_Handler
                        EXPORTPULSE_INIT

;-------------------------------------------------
My_Timer0A_Handler  PROC
                                        LDR     R1,=TIMER0_ICR; Clear the Flag
                                        MOV     R2,#0x04; Capture Mode Interrupt Clear
                                        STR     R2,[R1]
                                        LDR     R2,=GPIO_PORTF_DATA;
                                        LDR     R1,=TIMER0_TAILR;
                                        ;MOV    R0,#1;
                                        ;STR    R0,[R2]


                                        CMP R0,#1
                                        BEQ     LowCycle
                                        ;if R0 is 0
                                        LDR     R0,=HIGH
                                        STR     R0,[R1]; High Cycle Width loaded
                                        MOV     R0,#0x04;
                                        STR     R0,[R2]; Output set to high
                                        MOV     R0,#1
                                        BX LR
                                        ;if R0 is 1
LowCycle
                                        LDR     R0,=LOW
                                        STR     R0,[R1]; Low Cycle Width loaded
                                        MOV     R0,#0
                                        STR     R0,[R2]; Output set to low

                                        BX      LR
                                        ENDP
;-------------------------------------------------

PULSE_INIT      PROC
                        LDR R1, =SYSCTL_RCGCGPIO ; start GPIO clock
                        LDR R0, [R1]
                        ORR R0, R0, #0x20 ; set bit 5 for port F
                        STR R0, [R1]
                        NOP ; allow clock to settle
```

```
NOP
NOP
LDR R1, =GPIO_PORTF_DIR ; set direction of PF2
LDR R0, [R1]
ORR R0, R0, #0x04 ; set bit2 for output
STR R0, [R1]
LDR R1, =GPIO_PORTF_AFSEL ; regular port function
LDR R0, [R1]
BIC R0, R0, #0x04
STR R0, [R1]
LDR R1, =GPIO_PORTF_PCTL ; no alternate function
LDR R0, [R1]
BIC R0, R0, #0x00000F00
STR R0, [R1]
LDR R1, =GPIO_PORTF_AMSEL ; disable analog
MOV R0, #0
STR R0, [R1]
LDR R1, =GPIO_PORTF_DEN ; enable port digital
LDR R0, [R1]
ORR R0, R0, #0x04
STR R0, [R1]

LDR R1, =SYSCTL_RCGCTIMER ; Start Timer0
LDR R2, [R1]
ORR R2, R2, #0x01
STR R2, [R1]
NOP ; allow clock to settle
NOP
NOP
LDR R1, =TIMER0_CTL ; disable timer during setup LDR R2, [R1]
BIC R2, R2, #0x01
STR R2, [R1]
LDR R1, =TIMER0_CFG ; set 16 bit mode
MOV R2, #0x04
STR R2, [R1]
LDR R1, =TIMER0_TAMR
MOV R2, #0x02 ; set to periodic, count down
STR R2, [R1]
LDR R1, =TIMER0_TAILR ; initialize match clocks
LDR R2, =LOW
STR R2, [R1]
LDR R1, =TIMER0_TAPR
MOV R2, #15 ; divide clock by 16 to
STR R2, [R1] ; get 1us clocks
LDR R1, =TIMER0_IMR ; enable timeout interrupt
MOV R2, #0x01
STR R2, [R1]
; Configure interrupt priorities
```

```
; Timer0A is interrupt #19.
; Interrupts 16-19 are handled by NVIC register PRI4.
; Interrupt 19 is controlled by bits 31:29 of PRI4.
; set NVIC interrupt 19 to priority 2
                LDR R1, =NVIC_PRI4
                LDR R2, [R1]
                AND R2, R2, #0x00FFFFFF ; clear interrupt 19 priority
                ORR R2, R2, #0x40000000 ; set interrupt 19 priority to 2
                STR R2, [R1]
; NVIC has to be enabled
; Interrupts 0-31 are handled by NVIC register EN0
; Interrupt 19 is controlled by bit 19
; enable interrupt 19 in NVIC
                LDR R1, =NVIC_EN0
                MOVT R2, #0x08 ; set bit 19 to enable interrupt 19
                STR R2, [R1]
; Enable timer
                LDR R1, =TIMER0_CTL
                LDR R2, [R1]
                ORR R2, R2, #0x03 ; set bit0 to enable
                STR R2, [R1] ; and bit 1 to stall on debug
                BX LR ; return
                ENDP
                END
```

Main.s

```
;LABEL          DIRECTIVE       VALUE           COMMENT
                AREA    main, READONLY, CODE
                THUMB
                EXTERN          PULSE_INIT              ; Pulse initialization
                EXPORT          __main ; Make available

__main
                BL      PULSE_INIT; initialize pulse
                MOV     R0,#0; R0 is turn counter
loop            B       loop;

                ALIGN
                END
```

Q2)

Startup.s

Modified part is:

```
Timer0A_Handler
                        IMPORT      My_Timer0A_Handler
TIMER0A_Handler PROC
                        EXPORT  TIMER0A_Handler     [WEAK]
                        POP         {R0}
                        PUSH    {LR}
                        BL          My_Timer0A_Handler
                        POP         {LR}
                        PUSH    {R0}
                        BX          LR
                        ENDP
```

Read.s

```
; read.s
; Uses Timer0B to read pulse train on PB7


; 16/32 Timer Registers
TIMER1_CFG          EQU 0x40031000 ; Configuration Register
TIMER1_TAMR  EQU 0x40031004 ; Mode Register
TIMER1_CTL          EQU 0x4003100C ; Control Register
TIMER1_RIS          EQU 0x4003101C ; Raw interrupt Status
TIMER1_ICR          EQU 0x40031024 ; Interrupt Clear Register
TIMER1_TAILR  EQU 0x40031028 ; Interval Load Register
TIMER1_TAMATCHR EQU 0x40031030 ; Match Register
TIMER1_TAPR   EQU 0x40031038 ; Prescaling Divider
TIMER1_TAR          EQU 0x40031048 ; Counter Register
TIMER1_IMR          EQU     0x40031018 ; Defining Interrupt
TIMER1_TAV          EQU 0x40031050 ; To set the timer initial value

;GPIO Registers
;Port B base 0x40005000
GPIO_PORTB_IM               EQU 0x40005010 ; Interrupt Mask
GPIO_PORTB_DIR              EQU 0x40005400 ; Port Direction
GPIO_PORTB_AFSEL    EQU 0x40005420 ; Alt Function enable
GPIO_PORTB_DEN              EQU 0x4000551C ; Digital Enable
GPIO_PORTB_AMSEL    EQU 0x40005528 ; Analog enable
GPIO_PORTB_PCTL     EQU 0x4000552C ; Alternate Functions
GPIO_PORTB_PDR              EQU     0x40005514 ; Pull down
;System Registers
SYSCTL_RCGCGPIO     EQU 0x400FE608 ; GPIO Gate Control
SYSCTL_RCGCTIMER    EQU 0x400FE604 ; GPTM Gate Control

        AREA    routines, CODE, READONLY
```

```
                THUMB
                EXPORT          READ_INIT

READ_INIT       PROC
        LDR R1, =SYSCTL_RCGCGPIO ; start GPIO clock
        LDR R0, [R1]
        ORR R0, R0, #0x02 ; set bit 2 for port B
        STR R0, [R1]
        NOP ; allow clock to settle
        NOP
        NOP
        LDR R1, =GPIO_PORTB_DIR
        LDR R0, [R1]
        BIC R0, R0, #0x10 ; clear bit 4 for input
        STR R0, [R1]
; enable alternate function
        LDR R1, =GPIO_PORTB_AFSEL
        LDR R0, [R1]
        ORR R0, R0, #0x10 ; set bit4 for alternate fuction on PB4
        STR R0, [R1]
; set alternate function to T1CCP0 (7)
        LDR R1, =GPIO_PORTB_PCTL
        LDR R0, [R1]
        ORR R0, R0, #0x00070000 ; set bits 27:24 of PCTL to 7
        STR R0, [R1] ; to enable T1CCP0 on PB4
; disable analog
        LDR R1, =GPIO_PORTB_AMSEL
        MOV R0, #0 ; clear AMSEL to diable analog
        STR R0, [R1]
; Set pull down
        LDR     R1, =GPIO_PORTB_PDR
        MOV     R0, #0x10; set PB4 as pull down
        STR     R0, [R1];

;Timer1,A initialization
        LDR R1, =SYSCTL_RCGCTIMER
        LDR R2, [R1] ; Start timer 1
        ORR R2, R2, #0x02 ; Timer module = bit position (1)
        STR R2, [R1]
        NOP
        NOP
        NOP ; allow clock to settle
; disable timer during setup
        LDR R1, =TIMER1_CTL
        LDR R2, [R1]
        BIC R2, R2, #0x01 ; clear bit 0 to disable Timer 0
        STR R2, [R1]
; set to 16bit Timer Mode
```

```
        LDR R1, =TIMER1_CFG
        MOV R2, #0x04 ; set bits 2:0 to 0x04 for 16bit timer
        STR R2, [R1]
; set for edge time and capture mode
        LDR R1, =TIMER1_TAMR; set bit 4 to 0x01 for up counting
        MOV R2, #0x17 ; set bit2 to 0x01 for Edge Time Mode,
        STR R2, [R1] ; set bits 1:0 to 0x03 for Capture Mode
; set edge detection to both
        LDR R1, =TIMER1_CTL
        LDR R2, [R1]
        ORR R2, R2, #0x0C ; set bits 3:2 to 0x03
        STR R2, [R1]
; set start value
        LDR R1, =TIMER1_TAILR ; counter counts down,
        ;MOV R0, #0xFFFFFFFF ; so start counter at max value
        MOV R0, #0x00000000 ; so start counter at min value
        STR R0, [R1]
; Enable timer
        LDR R1, =TIMER1_CTL ;
        LDR R2, [R1] ;
        ORR R2, R2, #0x01 ; set bit 0 to enable
        STR R2, [R1]

                ENDP
                END
```

Main.s

```
; 16/32 Timer Registers
TIMER1_CFG      EQU 0x40031000 ; Configuration Register
TIMER1_TAMR     EQU 0x40031004 ; Mode Register
TIMER1_CTL      EQU 0x4003100C ; Control Register
TIMER1_RIS      EQU 0x4003101C ; Raw interrupt Status
TIMER1_ICR      EQU 0x40031024 ; Interrupt Clear Register
TIMER1_TAILR    EQU 0x40031028 ; Interval Load Register
TIMER1_TAMATCHR EQU 0x40031030 ; Match Register
TIMER1_TAPR     EQU 0x40031038 ; Prescaling Divider
TIMER1_TAR      EQU 0x40031048 ; Counter Register
TIMER1_IMR      EQU     0x40031018 ; Defining Interrupt
TIMER1_TAV      EQU 0x40031050 ; To set the timer initial value

FIRST           EQU     0x20000480
FREQ            EQU     0x00F42400 ; Freq 16M

;LABEL          DIRECTIVE       VALUE           COMMENT
                AREA    main, READONLY, CODE
                THUMB
                EXTERN          PULSE_INIT              ; Pulse initialization
                EXTERN          READ_INIT
```

```
                EXTERN      OutStr
                EXPORT      __main ; Make available

__main
                BL      READ_INIT;      initialize read
                BL      PULSE_INIT; initialize pulse
                MOV     R0,#0; R0 is turn counter
                PUSH{R0}

loop        LDR R1, =TIMER1_RIS
                LDR R2, [R1]
                ANDS R2,#04 ; isolate CAERIS bit
                BEQ loop ; if no capture, then loop

                LDR R1, =TIMER1_ICR;
                LDR R2, [R1];
                ORR     R2, #0x04; by setting CAECINT bit to 1, CAERIS bir is cleared

                LDR R1, =TIMER1_TAR ; address of timer register
                LDR R0, [R1] ; Get timer register value
                CMP     R3,#0
                MOVEQ       R4,R0; R4 0width        >-_-<
                ADDEQ R3,R3,#1; counter increased
                LDREQ  R1,      =TIMER1_TAV
                MOVEQ       R0,#0;
                STREQ  R0,      [R1];
                BEQ     loop;
                ; R3->1
                MOV     R5,R0; R5 Pwidth        >_-_<
                MOV     R3,#0; counter is set 0

                LDR     R1, =FIRST;
                ADD     R0,R5,R4; R0 is period in number
                MOV     R2,#625; a thick is 625*10^-10 sec
                MUL     R0,R0,R2; R0 is period in terms of 10^-10 sec
                STR     R0,[R1]; Period is written to memory
                ADD     R1,R1,#2; NEXT ADDRESS

                MUL     R0,R5,R2; R0 is pulse width in terms of 10^-10 sec
                STR     R0,[R1];
                ADD     R1,R1,#2; NEXT ADDRESS

                LDR     R2, =FREQ; R2 is 16M
                ADD     R0,R5,R4; R0 is period in number
                UDIV R0,R2,R0; R0 is frequency in Hz
                STR     R0,[R1];
                ADD     R1,R1,#2; NEXT ADDRESS
```

```
MOV     R2,#100;
ADD     R0,R5,R4; R0 is period in number
MUL     R5,R5,R2; R5 = 100*Pwidth
UDIV R0,R5,R0; D.C. cannot be calculated if <%1
STR     R0,[R1];
ADD     R1,R1,#2; NEXT ADDRESS

MOV     R2,#0x04;
STR     R2,[R1];
; all quantities in hex format
LDR     R5,=FIRST;
LDR     R1,     =TIMER1_TAV
MOV     R0,#0;
STR     R0,     [R1];
;PUSH{LR}
BL      OutStr
;POP{LR}
B       loop
ALIGN
END
```