EE 447-Introduction to Microprocessors

30.11.2016

Preliminary Work

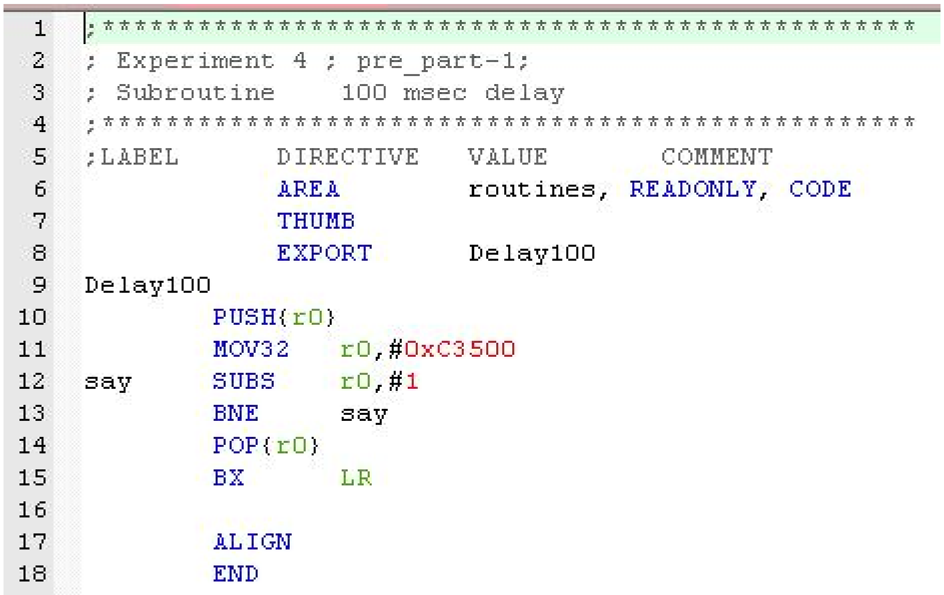
WA-1

Experiment-4

1) Cortex M4 in TM4C123GXL board run with 16 MHz system clock by default. To get 100 msec delay by using counter.

$$T = \frac{1}{f} = \frac{1}{16x10^6}\sec \text{ and required operation number } = \frac{100m\sec}{T} = 0.1x16x10^6 = 1600000$$

For hexadecimal value 1600000 = #0x186A00,`on the other hand we have two operation one of them decrements of counter and other comparison. Thus divide 186A00 by two and get #0xC3500. Fig. 1 shows the DELAY100 subroutine's assembly codes.

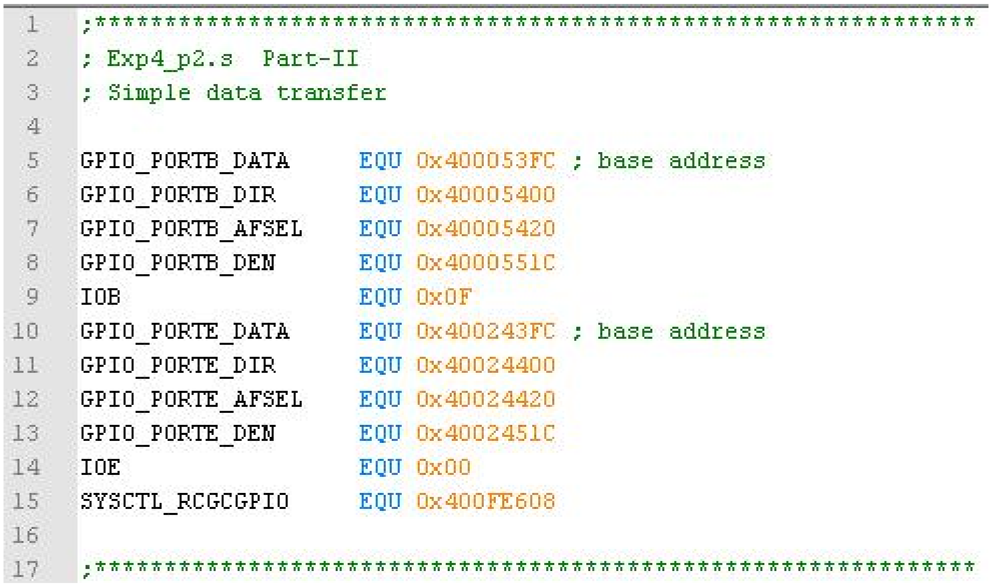


```
;**************************************************
; Experiment 4 ; pre_part-1;
; Subroutine    100 msec delay
;**************************************************
;LABEL      DIRECTIVE   VALUE       COMMENT
            AREA        routines, READONLY, CODE
            THUMB
            EXPORT      Delay100
Delay100
        PUSH{r0}
        MOV32   r0,#0xC3500
say     SUBS    r0,#1
        BNE     say
        POP{r0}
        BX      LR

        ALIGN
        END
```

Figure 1: DELAY100 subroutine assembly codes

2) Simple data transfer program's codes



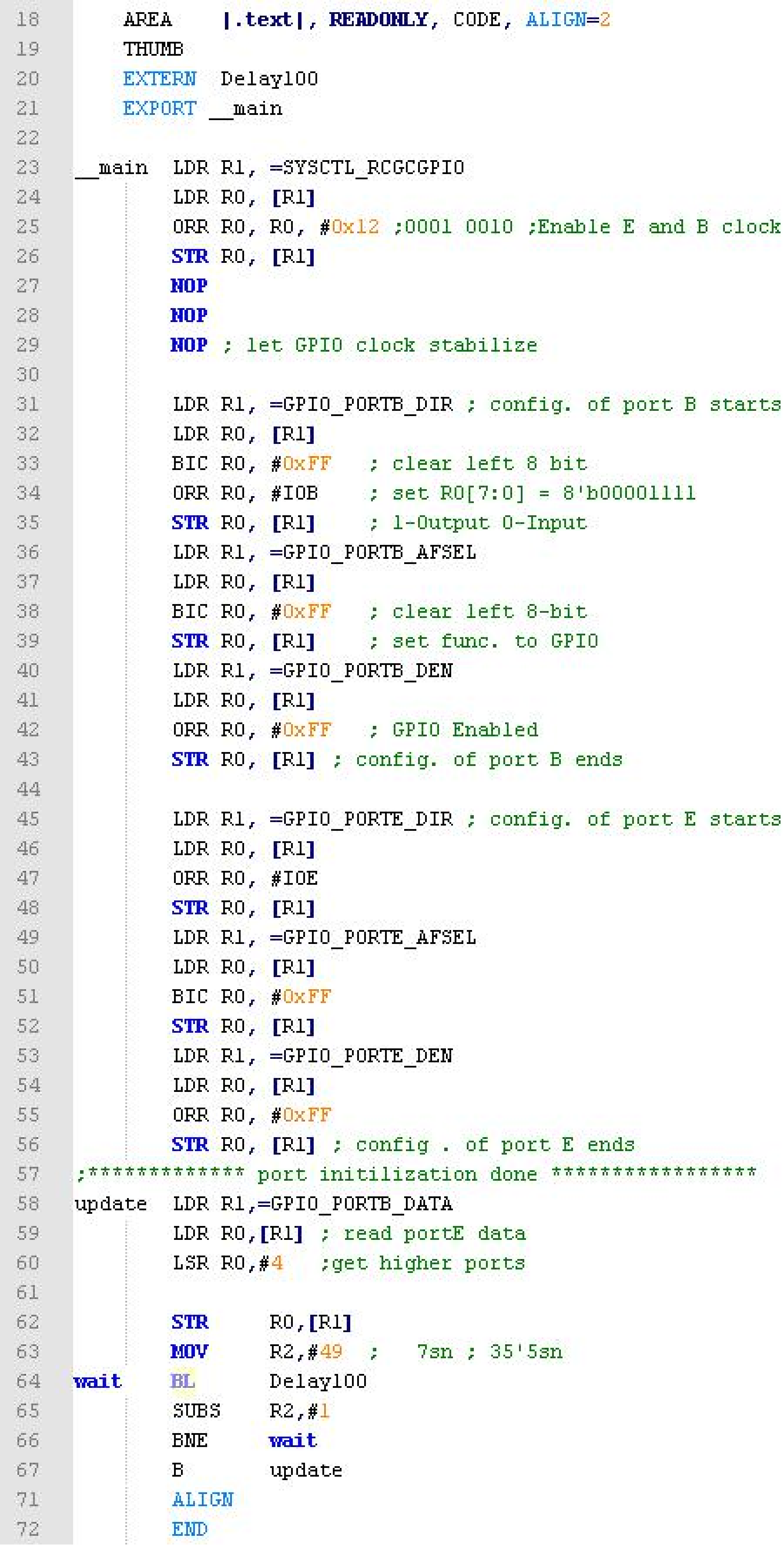
```
;*************************************************************
; Exp4_p2.s  Part-II
; Simple data transfer

GPIO_PORTB_DATA     EQU 0x400053FC ; base address
GPIO_PORTB_DIR      EQU 0x40005400
GPIO_PORTB_AFSEL    EQU 0x40005420
GPIO_PORTB_DEN      EQU 0x4000551C
IOB                 EQU 0x0F
GPIO_PORTE_DATA     EQU 0x400243FC ; base address
GPIO_PORTE_DIR      EQU 0x40024400
GPIO_PORTE_AFSEL    EQU 0x40024420
GPIO_PORTE_DEN      EQU 0x4002451C
IOE                 EQU 0x00
SYSCTL_RCGCGPIO     EQU 0x400FE608

;*************************************************************
```

EE 447-Introduction to Microprocessors

Preliminary Work



```
    AREA    |.text|, READONLY, CODE, ALIGN=2
    THUMB
    EXTERN  Delay100
    EXPORT __main

__main  LDR R1, =SYSCTL_RCGCGPIO
        LDR R0, [R1]
        ORR R0, R0, #0x12 ;0001 0010 ;Enable E and B clock
        STR R0, [R1]
        NOP
        NOP
        NOP ; let GPIO clock stabilize

        LDR R1, =GPIO_PORTB_DIR ; config. of port B starts
        LDR R0, [R1]
        BIC R0, #0xFF   ; clear left 8 bit
        ORR R0, #IOB    ; set R0[7:0] = 8'b00001111
        STR R0, [R1]    ; 1-Output 0-Input
        LDR R1, =GPIO_PORTB_AFSEL
        LDR R0, [R1]
        BIC R0, #0xFF   ; clear left 8-bit
        STR R0, [R1]    ; set func. to GPIO
        LDR R1, =GPIO_PORTB_DEN
        LDR R0, [R1]
        ORR R0, #0xFF   ; GPIO Enabled
        STR R0, [R1] ; config. of port B ends

        LDR R1, =GPIO_PORTE_DIR ; config. of port E starts
        LDR R0, [R1]
        ORR R0, #IOE
        STR R0, [R1]
        LDR R1, =GPIO_PORTE_AFSEL
        LDR R0, [R1]
        BIC R0, #0xFF
        STR R0, [R1]
        LDR R1, =GPIO_PORTE_DEN
        LDR R0, [R1]
        ORR R0, #0xFF
        STR R0, [R1] ; config . of port E ends
;************** port initilization done *****************
update  LDR R1,=GPIO_PORTB_DATA
        LDR R0,[R1] ; read portE data
        LSR R0,#4   ;get higher ports

        STR     R0,[R1]
        MOV     R2,#49  ;   7sn ; 35'5sn
wait    BL      Delay100
        SUBS    R2,#1
        BNE     wait
        B       update
        ALIGN
        END
```

EE 447-Introduction to Microprocessors

Preliminary Work

3) Keypad press detection program

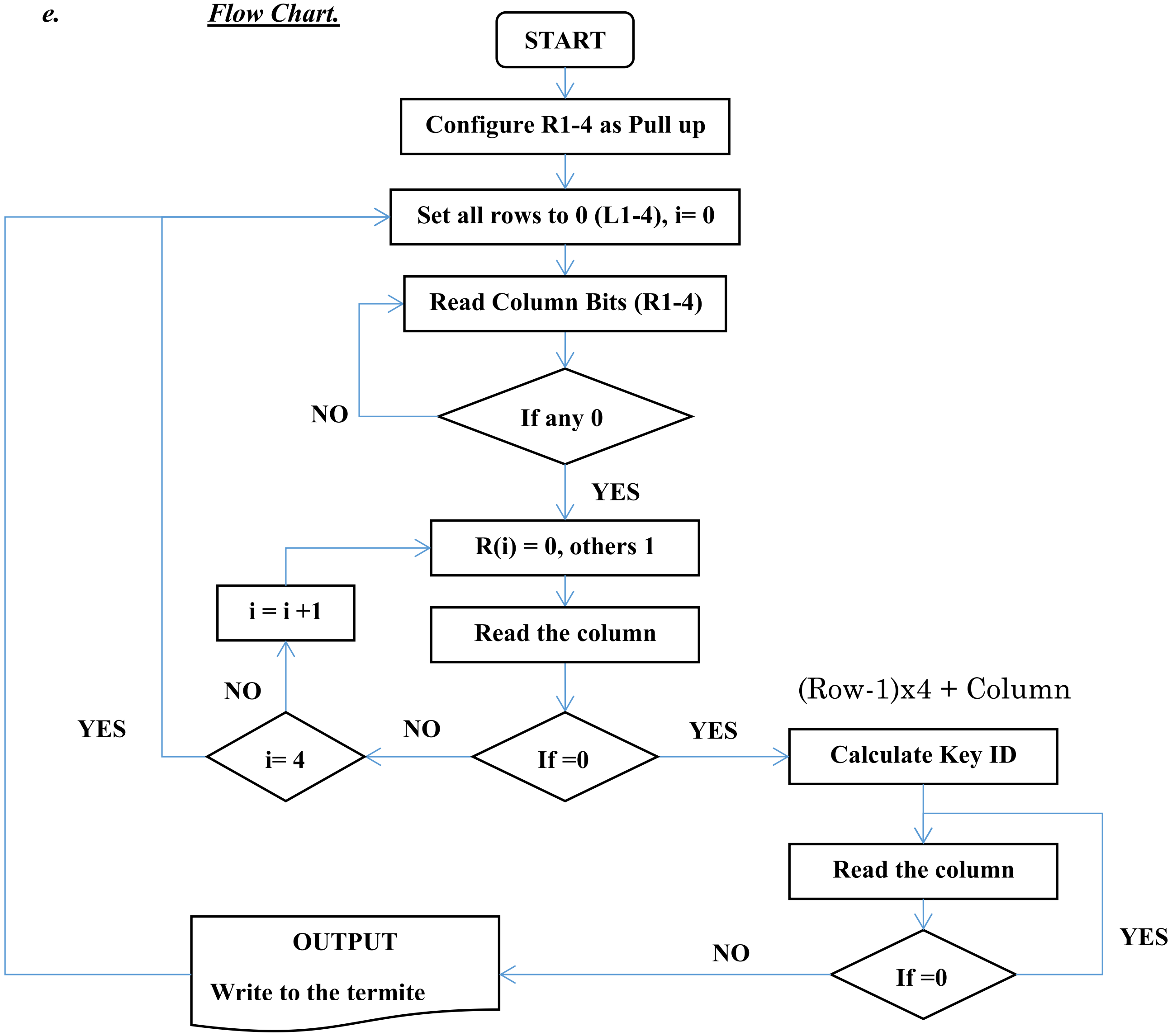
a. Set all columns (R1-4 pins) to pull up and output; all rows (L1-4 pins) as output low (all zeros), then read R1-4 status continuously. If one of the value of R1-4 is Low, this means that at least one key is pressed.

b. According to status of the column that include pressed key button, I contentiously check the status of corresponding pin. If it turn to one, release of the key can be detected.

c. To find the pressed key location, first I get column information from the key press detection step. Then, I start to scan rows sequentially to detect low voltage case. Then, accordingly row and column number corresponding ID information of related key can be calculated.

d. Bouncing may cause several press and release detection of a button at just one touching case. To be avoid that, I insert 100 msec delay just after the key press detection signal, while reading statutes of the keys and detecting pressed key.

e. ***Flow Chart.***



```
;****************************************************************
; Exp4_p3.s  Part-III
; Keypad press detection program

GPIO_PORTE_DATA     EQU 0x400243FC ; base address
GPIO_PORTE_ROW      EQU 0x400243C0 ; Rows   (L1-L4) Output
GPIO_PORTE_CLMN     EQU 0x4002403C ; Columns(R1-R4) Input
GPIO_PORTE_DIR      EQU 0x40024400
GPIO_PORTE_AFSEL    EQU 0x40024420
GPIO_PORTE_DEN      EQU 0x4002451C
GPIO_PORTE_PUR      EQU 0x40024510 ; PUR actual address
IOE                 EQU 0xF0; E7-4 output | E3-0 input
SYSCTL_RCGCGPIO     EQU 0x400FE608

;****************************************************************
    AREA    |.text|, READONLY, CODE, ALIGN=2
    THUMB
    EXTERN  Delay100
    EXTERN  OutStr
    EXTERN  CONVRT
    EXPORT __main

__main  LDR R1, =SYSCTL_RCGCGPIO
        LDR R0, [R1]
        ORR R0, R0, #0x10 ;0001 0000 ;Enable clock of port E
        STR R0, [R1]
        NOP
        NOP
        NOP ; let GPIO clock stabilize

        LDR R1, =GPIO_PORTE_DIR ; config. of port E starts
        LDR R0, [R1]
        BIC R0, #0xFF   ; clear 8 bit     Output     Input
        ORR R0, #IOE    ; b11110000  (L1-L2-L3-L4,R1-R2-R3-R4)
        STR R0, [R1]    ; 1-Out 0-In (E7,E6,E5,E4,E3,E2,E1,E0)
        LDR R1, =GPIO_PORTE_AFSEL
        LDR R0, [R1]
        BIC R0, #0xFF   ; clear left 8-bit
        STR R0, [R1]    ; set func. to GPIO
        LDR R1, =GPIO_PORTE_DEN
        LDR R0, [R1]
        ORR R0, #0xFF   ; GPIO Enabled
        STR R0, [R1]
        LDR R1,=GPIO_PORTE_PUR ; pull up E3-0(inputs)
        MOV R0,#0x0F    ;
        STR R0,[R1] ; config. of port E ends
;************* port initilization done *************************

        LDR R1,=GPIO_PORTE_ROW  ; L1-L4 |3C0
        LDR R2,=GPIO_PORTE_CLMN ; R1-R4 |03C
        mov r5,#4   ;constant
        ; Detect Pressed key
start   MOV R0,#0x00
        STR R0,[R1] ; set rows to 0
chk_1   LDR r3,[R2] ; read column data
        CMP r3,#0x0F
        BEQ chk_1   ;if all one stay in the check loop
        BL  Delay100
        mov r4,#0x70    ; rows 0111
        STR r4,[R1]     ;update rows
        LDR R0,[R2]
        CMP R0,#0x0F
        BNE calcID
        mov r4,#0xB0    ; rows 1011
        STR r4,[R1]     ;update rows
        LDR R0,[R2]
        CMP R0,#0x0F
        BNE calcID
        mov r4,#0xD0    ; rows 1101
        STR r4,[R1]     ;update rows
        LDR R0,[R2]
        CMP R0,#0x0F
        BNE calcID
        mov r4,#0xE0    ; rows 1110
        STR r4,[R1]     ;update rows
        LDR R0,[R2]
        CMP R0,#0x0F
        BEQ start       ; no press detect
    ;   r3-column data | r4-row data
calcID  LSRS r3,#5
        BHS  rs ; if carry 0 4th row else go 'rs'
        MOV  r3,#4  ; 4th row
        B    gec_1
rs      LSR  r3,#1  ; r3 show row number
gec_1   LSRS r4,#1
        BHS  cs ; if carry 0 4th column
        MOV  r4,#4  ; 4th row
        B    gec_2
cs      LSR  r4,#1  ; r3 show column number
gec_2   SUB  r3,#1
        MLA  R6,r3,r5,r4 ; get the key ID

        ; Detect Released key
chk_2   LDR r3,[R2] ; read column data
        CMP r3,#0x0F
        BNE chk_2   ;if key pressed stay in loop
        BL  CONVRT  ; out the R6 data to uart
        B   start

        ALIGN
        END
```