University of British Columbia
Electrical and Computer Engineering
Digital Systems and Microcomputers CPEN312

Lecture 13c: Writing 8051 Assembly

Dr. Jesús Calviño-Fraga
Department of Electrical and Computer Engineering, UBC
Office: KAIS 3024
E-mail: jesusc@ece.ubc.ca
Phone: (604)-827-5387

March 9, 2017

Step 1

```
$MODDE0CV ; Special Function Registers declaration for CV-8052

org 0000H ; After reset, the processor starts at location zero

Forever:
	cpl LEDRA.0 ; Turn LEDR0 on/off
	ljmp Forever; Repeat forever
END
```

Step 2

```
$MODDE0CV ; Special Function Registers declaration for CV-8052

org 0000H ; After reset, the processor starts at location zero

       mov LEDRA, #0 ; Turn off LEDs LEDR[0..7] Bit addressable
       mov LEDRB, #0 ; Turn off LEDs LEDR[8..9] Not bit addressable
Forever:
       cpl LEDRA.0 ; Turn LEDR0 on/off
       ljmp Forever     ; Repeat forever
END
```


Step 3

```
$MODDE0CV ; Special Function Registers declaration for CV-8052

org 0000H ; After reset, the processor starts at location zero

       mov LEDRA, #0 ; Turn off LEDs LEDR[0..7] Bit addressable
       mov LEDRB, #0 ; Turn off LEDs LEDR[8..9] Not bit addressable
Forever:
       cpl LEDRA.0 ; Turn LEDR0 on/off
       lcall Delay
       ljmp Forever     ; Repeat forever

Delay:
       mov R0, #250
L0:    djnz R0, L0 ; 3 machine cycles-> 3*30ns*250=22.5us
       ret
END
```

Step 4

```
$MODDE0CV ; Special Function Registers declaration for CV-8052

org 0000H ; After reset, the processor starts at location zero

       mov LEDRA, #0 ; Turn off LEDs LEDR[0..7] Bit addressable
       mov LEDRB, #0 ; Turn off LEDs LEDR[8..9] Not bit addressable
Forever:
       cpl LEDRA.0 ; Turn LEDR0 on/off
       lcall Delay
       ljmp Forever     ; Repeat forever

Delay:
       mov R2, #90
L2:    mov R1, #250
L1:    mov R0, #250
L0:    djnz R0, L0 ; 3 machine cycles-> 3*30ns*250=22.5us
       djnz R1, L1 ; 22.5us*250=5.625ms
       djnz R2, L2 ; 5.625ms*90=0.506s (approximately)
       ret
END
```


Step 5

```
$MODDE0CV ; Special Function Registers declaration for CV-8052

org 0000H ; After reset, the processor starts at location zero
       ljmp main
Delay:
       mov R2, #90
L2:    mov R1, #250
L1:    mov R0, #250
L0:    djnz R0, L0 ; 3 machine cycles-> 3*30ns*250=22.5us
       djnz R1, L1 ; 22.5us*250=5.625ms
       djnz R2, L2 ; 5.625ms*90=0.506s (approximately)
       ret

main:  mov sp, #0x7f ; Initialize stack pointer
       mov LEDRA, #0 ; Turn off LEDs LEDR[0..7] Bit addressable
       mov LEDRB, #0 ; Turn off LEDs LEDR[8..9] Not bit addressable
Forever:
       cpl LEDRA.0 ; Turn LEDR0 on/off
       lcall Delay
       ljmp Forever     ; Repeat forever
END
```

Step 6

```
main:
        mov SP, #0x7f
        mov LEDRA, #0 ; Bit addressable
        mov LEDRB, #0 ; Not bit addressable
Forever:
        mov HEX4, #0x61 ; Letter 'J' to HEX4
        lcall Delay
        mov HEX3, #0x06 ; Letter 'E' to HEX3
        lcall Delay
        mov HEX2, #0x12 ; Letter 'S' to HEX2
        lcall Delay
        mov HEX1, #0x41 ; Letter 'U' to HEX1
        lcall Delay
        mov HEX0, #0x12 ; Letter 'S' to HEX0
        lcall Delay
        ljmp Forever ; Repeat forever
END
```


Step 7

```
        mov HEX4, #0xff ; Clear HEX4
        mov HEX3, #0xff ; Clear HEX3
        mov HEX2, #0xff ; Clear HEX2
        mov HEX1, #0xff ; Clear HEX1
        mov HEX0, #0xff ; Clear HEX0
        lcall Delay
        lcall Delay
        ljmp Forever ; Repeat forever
END
```

Step 8

```
LETTER_J EQU #0x61
LETTER_E EQU #0x06
LETTER_S EQU #0x12
LETTER_U EQU #0x41
BLANK    EQU #0xff
.
.
.
mov HEX4, LETTER_J
lcall Delay
mov HEX3, LETTER_E
lcall Delay
mov HEX2, LETTER_S
lcall Delay
mov HEX1, LETTER_U
lcall Delay
mov HEX0, LETTER_S
lcall Delay
```


Step 9

```
        mov HEX5, BLANK
        mov HEX4, LETTER_J
        mov HEX3, LETTER_E
        mov HEX2, LETTER_S
        mov HEX1, LETTER_U
        mov HEX0, LETTER_S
Forever:
        lcall Delay
        mov R4, HEX5
        mov HEX5, HEX4
        mov HEX4, HEX3
        mov HEX3, HEX2
        mov HEX2, HEX1
        mov HEX1, HEX0
        mov HEX0, R4
        ljmp Forever ; Repeat forever
```

Step 10

```
Forever:
        lcall Delay
        jb SWA.0, Scroll_Right
        mov R4, HEX5
        mov HEX5, HEX4
        mov HEX4, HEX3
        mov HEX3, HEX2
        mov HEX2, HEX1
        mov HEX1, HEX0
        mov HEX0, R4
        ljmp Forever ; Repeat forever
Scroll_Right:
        mov R4, HEX0
        mov HEX0, HEX1
        mov HEX1, HEX2
        mov HEX2, HEX3
        mov HEX3, HEX4
        mov HEX4, HEX5
        mov HEX5, R4
        ljmp Forever ; Repeat forever
```


Step 11

```
; Look-up table for 7-seg displays.
T_7seg:
        DB 40H, 79H, 24H, 30H, 19H
        DB 12H, 02H, 78H, 00H, 10H
        DB 08H, 03H, 46H, 21H, 06H
        DB 0EH

Display_Number:
        mov dptr, #T_7seg
        mov a, R7
        anl a, #0x0f ; Force bits 4 to 7 to zero
        movc a, @dptr+a ; Read from table
        mov HEX0, a ; Display low nibble
        mov a, R7
        swap a ; exchange bits 0 to 3 with bits 4 to 7
        anl a, #0x0f ; Force bits 4 to 7 to zero
        movc a, @dptr+a ; Read from table
        mov HEX1, a ; Display high nibble
        ret
```

Step 11 (cont.)

```
Forever:
        mov R7, #0x00
        lcall Display_Number
        lcall Delay
        mov R7, #0x55
        lcall Display_Number
        lcall Delay
        mov R7, #0xAA
        lcall Display_Number
        lcall Delay
        ljmp Forever ; Repeat forever
```


Step 12

```
; Look-up table for 7-seg displays.
Show mac
        mov R7, %0
        lcall Display_Number
        lcall Delay
endmac

main:
        mov SP, #0x7f
        mov LEDRA, #0 ; Bit addressable
        mov LEDRB, #0 ; Not bit addressable
Forever:
        Show(#0x00)
        Show(#0x55)
        Show(#0xAA)
        ljmp Forever ; Repeat forever
END
```

Step 13

```
; Look-up table for 7-seg displays.
Show mac
       mov R7, %0
       lcall Display_Number
       lcall Delay
endmac

main:
       mov SP, #0x7f
       mov LEDRA, #0 ; Bit addressable
       mov LEDRB, #0 ; Not bit addressable
       mov R5, #0
Forever:
       Show(AR5) ; Use R5 first and explain why it fails
       inc R5
       ljmp Forever ; Repeat forever
END
```


Step 14

```
; Look-up table for 7-seg displays.
Show mac
       mov R7, %0
       lcall Display_Number
       lcall Delay
endmac

main:
       mov SP, #0x7f
       mov LEDRA, #0 ; Bit addressable
       mov LEDRB, #0 ; Not bit addressable
       mov R5, #0
Forever:
       Show(AR5)
       mov a, R5
       add a, #1
       da a
       mov R5, a
       ljmp Forever ; Repeat forever
END
```

Step 15

```
Display_at mac
        mov dptr, #T_7seg
        mov a, %2
        anl a, #0x0f ; Force bits 4 to 7 to zero
        movc a, @dptr+a ; Read from table
        mov %0, a ; Display low nibble
        mov a, %2
        swap a ; exchange bits 0 to 3 with bits 4 to 7
        anl a, #0x0f ; Force bits 4 to 7 to zero
        movc a, @dptr+a ; Read from table
        mov %1, a ; Display high nibble
endmac

Increment_BCD mac
        mov a, %0
        add a, #1
        da a
        mov %0, a
endmac
```


Step 15 (cont.)

```
        mov R3, #0x12
        mov R4, #0x59
        mov R5, #0x48
Forever:
        Display_at(HEX4, HEX5, R3)
        Display_at(HEX2, HEX3, R4)
        Display_at(HEX0, HEX1, R5)
        lcall Delay
        Increment_BCD(R5)
        cjne a, #0x60, Forever
        mov R5, #0
        Increment_BCD(R4)
        cjne a, #0x60, Forever
        mov R4, #0
        Increment_BCD(R3)
        cjne a, #0x13, Forever
        mov R3, #1
        ljmp Forever ; Repeat forever
END
```

Step 16

```
        jb KEY.3, skip_hour
        mov R3, SWA
skip_hour:
        jb KEY.2, skip_min
        mov R4, SWA
skip_min:
        jb KEY.1, skip_sec
        mov R5, SWA
skip_sec:

        mov a, SWB ; SWB is not bit addressable, but the acc is!
        jb acc.1, Forever ; Do not increment!
```