Assignment 2-2-3 Assembly Code

Assembly Code:

```
        .data
         cval:   .byte 'A', 'B', 'C'
         val1:   .word 0x12345678
         val3:   .word 257
         val2:   .asciiz "ABCD"
        .text
        .globl main
main:   lh $t0, val1    # $t0 is 5678
        lb $t1, val1    # $t1 is 78. # little endian system
```

Color Code = [red = old] [green = new] [yellow = highlighted segment]

My Observations:

First pass of

```
[00400024] lui $1, 4097                 ; 8: lh $t0, val1     # $t0 is 5678
```

Highlights the address at PC register = 0040024 and Changes

```
R31 (ra) = 00000000
R31 (ra) = 00400018
```

Then goes to/First pass of

```
[00400028] lh $8, 4($1)
```

Highlights the address at PC register = 0040028 and Changes

```
R1 (at) = 00000000
R1 (at) = 10010000
```

First pass of

```
[0040002c] lui $1, 4097          ; 9: lb $t1, val1     # $t1 is 78. # little
endian system
```

Highlights the address at PC register = 004002c and Changes

```
R8 (t0) = 00000000
R8 (t0) = 00005678
```

Then it goes to/First pass of

```
[00400030] lb $9, 4($1)
```

Highlights the address at PC register = 0040030

Followed by a change to $t0 at PC register = 00400034

```
R9 (t1) = 00000000
R9 (t1) = 00000078
```

End of Program

Elaborate the little endian system with these variable values

- **Val1**

Val1 = 0x12345678, with the little endian system 12 is MSB and 78 is LSB

(word) 0x12345678	**Memory Address**
0x78(LSB)	1001 0004
0x56	1001 0005
0x34	1001 0006
0x12(MSB)	1001 0007

- **$t0**

Assembly code states	**# in the Register $t0**
Lh $t0, val1 (load half byte of val 1)	00005678

The half word load instruction reads two bytes from the location of val1, because it's in little endian, it'll read and store the least significant half (5678) first.

- **$t1**

Assembly code states	**# in the Register $t2**
Lb $t1, val1 (load byte of val 1)	00000078

The byte load instruction reads one byte from the location of val1, because it's in little endian, it'll read and store the least significant byte (78) first.

- **Memory Address**
  - **1001 0000 / 1001 0004**

byte 'A','B','C'	**Memory Address**
A (0x41)	1001 0000
B (0x42)	1001 0001
C (0x43)	1001 0002

0 (empty)	1001 0003

- **1001 0008 / 1001 000C**

(word) int 275	**Memory Address**
0x01	1001 0008
0x01	1001 0009
0x00	1001 000a
0x00	1001 000b