CS 245 Fall 2017

Assignment 2 – Parsing an IPv6 Address

For all assignments, be sure to:

- Include file comments at the top, listing your name, the name of the program, and briefly describe what the program does.
- Describe the register convention, and calling sequence for procedures.
- Line up instruction parts in columns: 0=labels; 1tab=instruction mnemonic; 2tabs=operands; 3-4tabs=comments.
- Include pseudo-code comments to the right of your assembly code. **Avoid** comments that tell me what the assembly does, such as: move 25 to register $s3 – I already know this. Keep your comments to the logic.

Program 2: Parsing an IPv6 Address

In this assignment, you will take **an IPv6 Address stored in memory**, and print it in normal printed format.

An IPv6 (Internet Protocol version 6) address is used in data communications to define the sending and destination machine addresses. It is a 128-bit hexadecimal address that is printed by separating each 16 bits to form 8 parts, and printing each part in hexadecimal, separated by colons. Strings of zeros are converted to a single zero. See the example below.

Example: Defined in .data section:
```
Ipv6:   .word 0x58DC45C3 0x00000000 0x00000000 0x19abc912
```
Output:
```
    IPv6 Address: 58DC:45c3:0:0:0:0:19ab:c912
```
Notice that if 4 0 digits occur, they can be compressed into a single 0 digit. (More compressions exist, but this assignment only suggests implementing this one.)

Please use if statements, a loop, and indexed addressing (including `la` instruction and `lw $reg,0($reg)`. Print the headings ("IPv6 Address") and then the IP address in hexadecimal as shown.

I have provided starter code on my web site: www.cs.uwp.edu/Classes/Cs245. Notice that there is a procedure you can call to print each character. You send it a nibble and it will print the ASCII character.

To get your logic correct, why not write this in java or pseudocode then write the assembly for it?

Submission

Turn this program in as assign2.asm **via paper and electronic copy**:

```
$ submit 245 homework2.asm
```

Grading

Each homework assignment is worth 10 points. Be careful to include all comments and format correctly, as directed in the beginning of this file.

Assignment 3: Parsing an IPv6 message

In this third homework assignment you will parse an IPv6 packet header. The fields are allocated as shown in the following diagram:

Version (4)	Diff Serv (6)	ECN (2)	Flow Label (20 bits)
Payload Length (16)		Next Header (8)	Hop Limit (8)
Source Address (120 bits)			
Destination Address (120 bits)			

Not that you need to know this, but just for your interest, those fields are defined as follows:

- Version (4 bits): Generally a 4 or 6 to indicate the version of the IP protocol.
- Differentiated Services (DS) (6 bits): Describes the specific service; used for specialized routing.
- Explicit Congestion Notification (2 bits): Values: 00=Not supported; 01 or 10=Supported; 11=Congestion.
- Flow Label (20 bits): Labels packets with special handling, e.g., with quality of service requirements.
- Payload Length (16 bits): Length of the remainder of the packet not including the header in bytes.
- Next Header (8 bits): Identifies the next protocol header that follows this IPv6 protocol header. Values: TCP=6, UDP=17.
- Hop Limit (8 bits): The remaining number of hops that this packet is allowed.
- Source Address (128 bits): Source IPv6 address: Sender of packet
- Destination Address (128 bits): Destination IPv6 address: Receiver of packet

Your job is to print out a packet header based on the protocol layout, extending homework 2.

```
Example input: 0x61 30 00 2a   00 61 11 01   fe 80 00 00   00 00 00 00   45 3e e2 91
               38 8d ed f1   ff 02 00 00   00 00 00 00   00 00 00 00   00 01 00 02
               Note: 0x6 is the version in first word.
```

Example data: (Does not match input)

```
    IP Version 6 Data:
    Version: 6      DiffServ: 0          ECN: 0x3      Flow Label: 0x04503
    Payload Length: 1200       Next Hdr: 6          Hop Limit: 125
    Source Address: 0:0:0:0:0:0:a34b:23f4
    Destination Address: 0:0:0:0:0:0:a34b:2146
```

You will need procedures in this assignment. Please use these procedure headers:

```
######################################################################
#  Print_name_decimal($a0=address of ascii name, $a1=value to print in decimal)
#  Prints a label and its value in decimal.
#  Example output: Payload Length: 1200
#  Calling Sequence:
#       la      $a0,Label       # Print_name_decimal($a0=ASCII addr, $a1=decimal value)
#       lw      $a1,Value
#       jal     PrNmDec
######################################################################

######################################################################
#  Print_IP_addr(address of ascii name, address of IP address)
#  Prints a label and an IPv6 address.  This program calls Parse_IP_addr(IP address)
#  Example output: Destination Address: 0:0:0:0:0:0:a34b:2146
#  Calling Sequence:
#       la      $t1,Label       # Print_IP_addr(address of ascii name, address of IP address)
#       sw      $t1,-4($sp)
#       la      $t1,ipaddr
#       sw      $t1,-8($sp)
#       jal     PrIpAd
#######################################################################

######################################################################
#  Parse_IP_addr(address of IP address)
#  Prints an IPv6 address given an address of an IPv6 address.  (Basically this is homework 2)
#  Example output: 2436:0:0:0:0:0:a34b:2146
#  Calling Sequence:
#       la      $t1,ipaddr      # Parse_IP_addr(address of IP address)
#       sw      $t1,-4($sp)
#       jal     ParsIPad
#######################################################################

######################################################################
#  Print_name_hex($a0=address of ascii name, $a1=value to print in decimal)
#  Prints a label and its value in hexadecimal.
#  Example output: Flow Label: 0x04503
#  Calling Sequence:
#       la      $a0,Label       # Print_name_hex($a0=ascii address, $a1=hex value to print)
#       lw      $a1,Value
#       jal     PrNmHex
######################################################################
```

Requirements:
Please use the procedures and calling sequence as specified in the comments above. Use stacks minimally to save off the $ra. Avoid using $s registers in your procedures, or if you do, save them off on the stack and reload them.

You can copy the header comments and use them directly.

Submission

Turn this program in as homework3.asm **via paper and electronic copy**:

```
$ submit 245 homework3.asm
```

Grading

Each homework assignment is worth 10 points. Be careful to include all comments and format correctly, as directed in the beginning of this file.