Q4. Write an ARM assembly program that checks if a 32-bit number is a palindrome. Assume that the input is available in r 3. The program should set r 4 to 1 if it is a palindrome, otherwise r 4 should have 0. A palindrome is a number which is the same when read from both sides. For example, 1001 is a 4 bit palindrome.

Solution :

```
mov R0,1044480
mov R3,0
mov R4,31
movu R2,0x0001
mov R5,0
andu R1,R0,0xffff

.loop:
and R3,R2,R1
lsl R2,R2,1
lsl R3,R3,R4
add R5,R5,R3
sub R4,R4,2
cmp R4,0
bgt .loop
lsr R0,R0,16
lsr R5,R5,16
sub R0,R5,R0
cmp R0,0
beq .palin
mov R0,0

b .exit

.palin:
mov R0,1

.exit:
```

Q6. Consider a direct mapped cache with 16 cache lines, indexed 0 to 15, where each cache line can contain 32 integers (block size : 128 bytes). Consider a two-dimensional, 32*32 array of integers a . This array is laid out in memory so that a [0; 0] is next to a [0; 1], and so on. Assume the cache is initially empty, but that a [0; 0] maps to the first word of cache line 0.
Consider the following column first traversal:

```
{
 int sum = 0;
for (int i = 0; i < 32; i++) {
for( int j=0; j < 32; j++) {
sum += a[i,j];
}
}
```

and the following row-rst traversal:

```
{
int sum = 0;
for (int i = 0; i < 32; i++) {
for( int j=0; j < 32; j++) {
 sum += a[j,i];
}
}
```

Compare the number of cache misses produced by the two traversals, assuming
the oldest cache line is evicted first. Assume that i , j , and sum are stored in registers. Assume that no part of array, a , is saved in registers. It is always stored in the cache.

Solution :

Number of cache misses in column first traversal = 32. Miss rate = 3.1%.
Number of cache misses in row rst traversal = 32* 32 = 1024. Miss rate = 100%.

Q2. You have 3 cache designs for a 16-bit address machine.

Slytherin	Gryffindor	Ravenclaw
Direct-mapped cache. Each cache line is 1 byte. 10-bit index, 6-bit tag. 1 cycle hit time.	2-way set associative cache. Each cache line is 1 word (4 bytes). 7-bit index, 7-bit tag. 2 cycle hit time.	fully associative cache with 256 cache lines. Each cache line is 1 word. 14-bit tag. 5 cycle hit time.

(a) What is the size of each cache?

(b) How much space does each cache need to store tags?

(c) Which cache design has the most con ict misses? Which has the least?

(d) If someone told you the hit rate for the 3 caches is 50%, 70% and 90% but did not tell you which hit rate corresponds to which cache, which cache would you guess corresponded to which hit rate? Why?

Solution :

(a) What is the size of each cache?

Slytherin: 1024 1B lines = 1KB.
Gryffindor: 128 4B lines * 2 ways = 1KB
Ravenclaw: 256 4B lines = 1KB

(b) How much space does each cache need to store tags?
Slytherin: 1024 x 6-bit tags = 6Kb
Gryffindor: 256 x 7-bit tags = 1792 bits
Ravenclaw: 256 x 14-bit tags = 3584 bits

(c) Slytherin probably has the most con ict misses, since it is direct mapped. Ravenclaw, because it is fully associative, can never have conflict misses.

(d) Since the caches are the same size and the reason in the previous answer, Slytherin is 50%, Gryffindor is 70%, and Ravenclaw is 90%.

Q3. The 5 stages of the processor have the following latencies:

	Fetch	Decode	Execute	Memory	Writeback
a.	300ps	400ps	350ps	550ps	100ps
b.	200ps	150ps	100ps	190ps	140ps

Assume that when pipelining, each pipeline stage costs 20ps extra for the registers between pipeline stages.

a. **Non-pipelined processor:** what is the cycle time? What is the latency of an instruction? What is the throughput?

Solution : As there is no pipelining, the cycle time must allow an instruction to go through all stages in one cycle. The latency is the same as cycle time since it takes the instruction one cycle to go from the beginning of fetch to the end of writeback. The throughput is defined as **1/CT** inst/s.
a. CT = 300 + 400 + 350 + 550 + 100 = 1700ps
Latency = 1700ps
Throughput = 1/1700 inst/ps
b. CT = 200 + 150 + 100 + 190 + 140 = 780ps
Latency = 780ps
Throughput = 1/780 inst/ps

The latency for an instruction is also the same, since each instruction takes 1 cycle to go from beginning fetch to the end of writeback. The throughput similarly is 1/cycle time instructions per second.

b. **Pipelined processor:** What is the cycle time? What is the latency of an instruction? What is the throughput?
Solution : Pipelining reduces the cycle time to the length of the longest stage plus the register delay. Latency becomes CT*N where N is the number of stages as one instruction will need to go through each of the stages and each stage takes one cycle. The throughput formula remains the same.
a. CT = 550 + 20 = 570 ps
*Latency = 5 * 570 = 2850ps*
Throughput = 1/570 inst/ps
b. CT = 200 + 20 = 220 ps
*Latency = 5 * 220 = 1100ps*
Throughput = 1/220 inst/ps

c. If you could split one of the pipeline stages into 2 equal halves, which one would you choose? What is the new cycle time? What is the new latency? What is the new throughput?
Solution : We would want to choose the longest stage to split in half. The new cycle time becomes theoriginally 2nd longest stage length. Calculate latency and throughput correspondingly, but remember there are now 6 stages instead of 5.
a. CT = 400 + 20 = 420 ps
Latency = 6 * 420 = 2520 ps
Throughput = 1/420 inst/ps
b. CT = 190 + 20 = 210 ps
Latency = 6 * 210 = 1260 ps

Throughput = 1/210 inst/ps

Q5 (a). Consider a fully associative cache following the LRU replacement scheme and consisting of only 8 words. Consider the following sequence of memory accesses (the numbers denote the word address):
20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 22, 30, 21, 23, 31
Assume that we begin when the cache is empty. What are the contents of the cache after the end of the sequence of memory accesses.

Solution :

28,29,30,21,23,31,26,27

Q5 (b). Number of the times the instruction sequence below will loop before coming out of loop is ----------

```
mov r1, 00h
A1: add r1,r1,1
jnz A1
```

Solution : 256

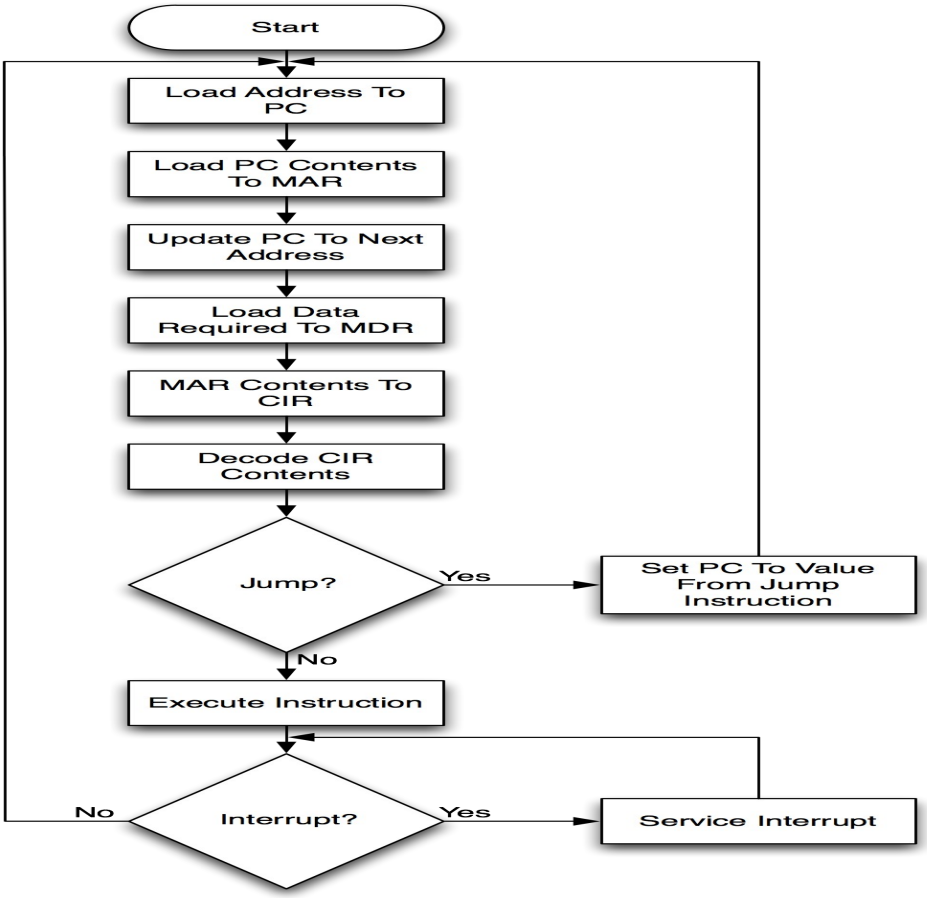
Justification : r1 = 0000 0000. The loop runs from 00000000 to 11111111 = 256 times after which the r1 becomes 00000000 again and the it exits.

Q5 (c).

CISC	**RISC**
Emphasis on hardware	Emphasis on software
Includes multi-clock complex instructions	Single-clock,reduced instruction only
Memory-to-memory: "LOAD" and "STORE" incorporated in instructions	Register to register: "LOAD" and "STORE"a re independent instructions
Small code sizes, high cycles per second	large code sizes, Low cycles per second
Transistors used for storing complex instructions	Spends more transistors on memory registers

Q1. **Solution :**

Instruction cycle :



(b) Why interrupt flag is used?

Solution :

Each individual interrupt source has its own interrupt enable bit. This allows the programmer to enable only the interrupts that are needed for a particular system. For example, the serial communications port can be set up to generate an interrupt only when a byte of data is received. This is done by setting an interrupt enable bit for the communications receive.

The I flag is a global interrupt enable/disable bit. All of the interrupt sources are gated with the I flag. If the I flag is set, none of the interrupts will be seen by the processor hardware. This allows the programmer to easily disable/enable all interrupts.

If a particular section of code is time sensitive, it may be necessary to disable all interrupts while executing that code in order to prevent an interrupt from slowing the execution. To do this, an SEI instruction is placed before the time sensitive code and a CLI instruction is placed after the time sensitive code to make sure no interrupts occur in the time sensitive code. The interrupt response time can be adversely affected when interrupts are disabled.