1. ابتدا تعداد بیت های آدرس را مشخص میکنیم. از آنجا که حافظه *16MB* و *Byte-Addressable* است، پس آدرس ما $\log(16 \times 2^{20}) = 24$ بیتی است.

آ) *direct mapped*:

$$\#offset\ bits = \log(block\ size) = \log 64 = 6$$
$$\#sets = cash\ size / cash\ block\ size = \frac{6 \times 2^{10}}{64} = 1024$$
$$\#index\ bits = \log(\#sets) = 10$$
$$\#tag\ bits = 24 - 6 - 10 = 8$$

2-way associative :

$$\#offset\ bits = \log(block\ size) = \log 64 = 6$$
$$\#sets = cash\ size / cash\ block\ size = \frac{6 \times 2^{10}}{64 \times 2} = 512$$
$$\#index\ bits = \log(\#sets) = 9$$
$$\#tag\ bits = 24 - 6 - 9 = 9$$

ب) کافیست با فرمول محاسبه کنیم (تعداد کلاک ها در *Hit Time* کش سطح اول را همان *CPI* پایه در نظر میگیریم)

$$AMAT = Hit\ Time + (Miss\ Rate \times Miss\ Penalty) = 1.5 + \big(0.08 \times (20 + 0.02 \times 100)\big)$$
$$\Rightarrow CPI = 0.6 \times 1.5 + 0.4 \times \Big(1.5 + \big(0.08 \times (20 + 0.02 \times 100)\big)\Big) = 2.204$$

2. آ) فرض میکنیم که سیستم ما *Byte Addressable* است.از آنجایی که $\#offset\ bits = \log(block\ size)$ پس هر بلاک $2^5 =$ 32 بایت است که معادل *8* کلمه *32* بیتی است.

ب) مجددا از آنجایی که $\#index\ bits = \log(\#sets)$ و $cache\ entries = set\ counts \times associativity$ پس $2^5 =$ 32 درایه داریم.

ج) در هر خط، یک بلاک داده معادل *8×32*بیت داده داریم. *22* بیت *tag* و *1*بیت *valid*نیز در هر *cache entry* داریم. (میتوان *dirty bit,* و.. *status bit* را نیز به همین منوال در نظر گرفت) پس در کل: $\frac{256}{256+22+1} = 91.75\%$

د)

Miss/Hit	*Set number (index in demical)*	*Offset bits*	*Index bits*	*Tag bits[2:0]*	*Binary*	*Address*
Miss	*0*	*00000*	*00000*	*000*	*000000000000*	*0*
Hit	*0*	*00100*	*00000*	*000*	*000000000100*	*4*
Hit	*0*	*10000*	*00000*	*000*	*000000010000*	*16*
Miss	*4*	*00100*	*00100*	*000*	*000010000100*	*132*
Miss	*7*	*01000*	*00111*	*000*	*000011101000*	*232*
Miss	*5*	*00000*	*00101*	*000*	*000010100000*	*160*
Miss	*0*	*00000*	*00000*	*001*	*010000000000*	*1024*
Miss	*0*	*11110*	*00000*	*000*	*000000011110*	*30*
Hit	*4*	*01100*	*00100*	*000*	*000010001100*	*140*
Miss	*0*	*11100*	*00000*	*011*	*110000011100*	*3100*
Hit	*5*	*10100*	*00101*	*000*	*000010110100*	*180*
Miss	*4*	*00100*	*00100*	*010*	*100010000100*	*2180*

ه) نرخ برخورد معادل تعداد *Hit* ها به کل درخواست ها برابر $\frac{4}{12} = 33.33\%$ است.

3. ابتدا مانند سوالات قبل بیت هارا محاسبه میکنیم. $\log 32 = 5$ بیت برای آفست، $\log 8 = 3$ برای ایندکس و $(\log 64 \times 1024) - 3 - 5 = 8$ بیت برای برچسب میخواهیم. حال کافیست با منطق *little endian* هر بایت اشاره شده را پشت باید بعدیش بگذاریم تا به کلمه اشاره شده برسیم.

Hit / Miss	Set number (index in demical)	Offset bits	Index bits	Tag bits	Address
Miss	0	00000	000	00000000	0x0000
Miss	1	10010	001	01001010	0x4A32
Hit	0	00010	000	00000000	0x0002
Miss	1	10000	001	10111000	0xB830
Hit	0	00100	000	00000000	0x0004
Hit	1	10010	001	10111000	0xB832
Hit	0	00110	000	00000110	0x0006
Hit	1	10100	001	10111000	0xB834
Hit	0	01000	000	00000000	0x0008
Hit	1	00010	001	10111000	0xB822
Hit	0	01010	000	00000000	0x000A
Miss	1	10010	001	01001010	0x4A32
Hit	0	01100	000	00000000	0x000C
Miss	0	11010	000	00110001	0x311A
Miss	0	01110	000	00000000	0x000E
Miss	2	00010	010	00110001	0x3142

نرخ برخورد معادل تعداد *Hit* ها به کل درخواست ها و برابر $\frac{9}{16} = 56.25\%$ است.

4. آ) هرکدام از 6 دستور درون حلقه باید از حافظه *fetch* شوند و دو دستور *lb* و*sb* نیز با حافظه کار میکنند. پس سر جمع در هر اجرا 8 بار کار با حافظه داریم.

ب) این بار تنها 3 دستور ابتدایی حلقه اجرا میشوند که نیازمند 3 بار واکشی دستور (فرض کردیم بار اول است که این حلقه اجرا میشوند و نیاز به حافظه اصلی داریم) و 2 بار کار با حافظه توسط *lb, sb* هستند. پس سر جمع 5 دسترسی به حافظه دارد.

طبق صورت سوال هر بلوک 1 کلمه یا معادلا در زبان میپس 4 بایت است. بنابراین حافظه نهان ما 32 تا *set* دارد که برای نمایش آنها به 5 بیت برای *index* نیاز داریم. همچنین چون هر بلوک 4 بایت است نیازی به 2 بیت *offset* داریم.

Set number	Offset(2bits)	Index(5bits)	Address	I/D
0	00	00000	0x0000	lb $t2, 0($t0)
1	00	00001	0x0004	sb $t2, 0($t1)
2	00	00010	0x0008	beq $t2, $zero, Exit
3	00	00011	0x000C	addiu $t0, $t0, 1
4	00	00100	0x0010	addiu $t1, $t1, 1
5	00	00101	0x0014	j StrCpy
0	00-11	00000	0x0080-0x0083	T0
0	00-11	00000	0x0100-0x0103	T1

ج) اگر منظور از حافظه را مجموعه ی حافظه اصلی و حافظه نهان در نظر بگیریم 3*8+5 برابر با 29 جواب ماست.

اما اگر منظور از حافظه را حافظه اصلی سیستم در نظر بگیریم. درین صورت در اولین بار اجرای حلقه 6 دستور ما درون کش قرار میگیرند و بجز دستور اول در حلقه های بعدی نیازی به واکشی بقیه دستورات نیست پس حلقه های دیگر 2+1 دسترسی به حافظه دارند. پس سر جمع 8 + 3*3 =17 دسترسی به حافظه داریم.

د) طبق فرض در *cache* از سیاست *no-write-allocate* استفاده میکنیم. نوشتن در حافظه را جزء دسترسی ها به کش حساب نمیکنیم (پس متناسب با ج در کل 29-4=25 دسترسی به کش داریم) . بنابر این در حلقه اول 7 بار میس (6 دستوری + یک دیتا) و در 3 حلقه بعدی هرکدام 2 بار میس (دستور اول + یک دیتا) داریم. پس در کل 13 میس داریم و:

$$miss\ rate = \frac{13}{25} = 52\%$$

ه) هر *set* کش دارای دو بلوک 4بایتی است. بنابراین دستور *lb* و داده حاصل از خواندن از ممووری میتوانند همزمان در کش قرار بگیرند. در نتیجه صرفا در اولین بار اجرای حلقه *Miss* میخوریم و در دفعات دیگر اجرای حلقه فقط *Hit* میخوریم:

$$miss\ rate = \frac{7}{25} = 28\%$$

و) این بار دیگر دستور نوشتن از حافظه جز دسترسی های کش محسوب میشود. پس این بار کل دسترسی های ما به مجموعه حافظه برابر دسترسی های ما به کش است که در ابتدای ج برابر با 29 (25+4) محاسبه اش کردم. حال این بار به میس های قبلی، نوشتن در حافظه نیز اضافه میشود. پس :

$$miss\ rate = \frac{13+4}{29} = 58.6\%$$

5. آ)

- تعداد بلاک ها در حافظه: $\#ways \times 2^{\#index\ bits} = 2 \times 2^{10} = 2048$
- تعداد *Block offset*: $\log 32 = 5$
- اندازه *Tag*: $32 - 10 - 5 = 17$

ب) فرض میکنیم به صورت *Fully associative* است. درین صورت داریم:

Hit / Miss	Cache status	Offset bits	Tag bits	Address
Miss	C[0]=[110000:110011]	01	1100	110001
Miss	C[1]=[100100:100111]	11	1001	100111
Miss	C[2]=[001100:001111]	11	0011	001111
Hit	C[2]	00	0011	001100
Miss	C[3]=[010000:010011]	01	0100	010001
Hit	C[0]	10	1100	110010
Hit	C[1]	01	1001	100101
Hit	C[2]	10	0011	001110
Miss	C[3]=[100000:100011]	01	1000	100001
Hit	C[0]	01	1100	110001

ج) مشابه بخش ب سوال اول عمل میکنیم. میدانیم در *pipline* از کلاک پنجم به بعد، درصورتی که با خطا مواجه نشویم در هر کلاک یک دستور کامل میشود. دستوراتی که از حافظه استفاده میکنند، *load, store* هستند که به طور میانگین و با محاسبه خطا تعداد کلاک هایشان از فرمول زیر بدست می آید.

$$AMAT = Hit\ Time + (Miss\ Rate \times Miss\ Penalty) = 1 + \big((1 - 0.98) \times 100\big) = 3$$

بنا برین این دو دستوربه طور میانگین 2 کلاک اضافه تر میخواهد و 2 کلاک *stall* میخوریم. پس داریم:

$$Execution\ Time = 4 + 1 \times IC + 2 \times 0.4 \times IC \overset{IC\gg}{\Longrightarrow} CPI \approx \frac{1.8\ IC}{IC} = 1.8$$