



Nome: \_\_\_\_\_

**Notas importantes!**

1. Verifique, para todas as questões, qual a resposta correta e anote-a com uma cruz (x) na tabela ao lado. Por cada resposta incorreta será descontada, à cotação global, no máximo 1/3 da cotação da respectiva pergunta.
2. Pode usar até um máximo de 4 respostas duplas (por cada dupla, 0 acertos desconta um 2/3, 1 acerto conta um 7/9). Se usar mais de 4 duplas, serão aceites as 4 primeiras e as restantes serão consideradas respostas erradas.
3. Durante a realização do teste não é permitida a permanência junto do aluno, mesmo que disponibilizado, de qualquer dispositivo eletrónico não expressamente autorizado (esta lista incluem-se calculadoras, telemóveis e smartwatch). A sua deteção durante a realização do exame implica a imediata anulação do mesmo.

**Grupo I**

1. Considere que no endereço de memória `0x00400020` se encontra armazenada a instrução `beq $6, $2, label`. Se `label` corresponde ao endereço `0x00400000`, o valor dos 16 bits menos significativos do código máquina dessa instrução será:
  - a. `0x0FFF`
  - b. `0x1FFF`
  - c. `0x2FFF`
  - d. `0x3FFF`
2. Suponha que `$3=0x0FC4756EB` e `$2=0x00FFFF00`. A instrução `"xor $6, $3, $2"` produz o seguinte resultado armazenado em `$6`:
  - a. `0x3047A9EB`
  - b. `0x0FC0000EB`
  - c. `0x0FC88A9EB`
  - d. `0x00B80017`
3. Na arquitetura básica de um sistema computacional o *Control Bus* permite:
  - a. transferir dados entre os registos do CPU.
  - b. especificar a natureza de uma operação efetuada sobre a memória.
  - c. transferir o código máquina das instruções para o *instruction register*.
  - d. identificar, na memória, a origem/destino dos dados transferidos.
4. A arquitetura MIPS não permite realizar operações lógicas e/ou aritméticas sobre o conteúdo de posições da memória externa, ou, simultaneamente, sobre o conteúdo de registos internos e posições da memória externa. Por esse razão:
  - a. permite, para a mesma instrução e num mesmo ciclo de relógio, operar sobre a ALU e efetuar uma operação de leitura da memória.
  - b. todas as variáveis de um dado programa devem residir em registos internos.
  - c. é menos eficiente do que arquiteturas que permitem esse tipo de operações.
  - d. é considerada como sendo uma arquitetura do tipo *Load-Store*.
5. Um endereço de memória externa num sistema computacional é:
  - a. a informação armazenada em cada posição de memória.
  - b. um número único que identifica cada posição de memória.
  - c. o valor transferido de um registo do CPU para uma posição de memória.
  - d. o meio de identificar o tipo de operação realizada sobre essa memória.
6. Os processadores da arquitetura hipotética ZXIoT implementam um total de 130 instruções. Todas as instruções são codificadas com 32 bits, e um dos formatos tem 5 campos: *opcode*, três campos para identificar registos internos e um campo para codificar constantes no intervalo  $[-16384, +16383]$ . Considerando que o número de bits do campo *opcode* é apenas o necessário para codificar as 130 instruções, conclui-se que o número de registos é:
  - a. 32
  - b. 16
  - c. 8
  - d. 4

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255	256	257	258	259	260	261	262	263	264	265	266	267	268	269	270	271	272	273	274	275	276	277	278	279	280	281	282	283	284	285	286	287	288	289	290	291	292	293	294	295	296	297	298	299	300	301	302	303	304	305	306	307	308	309	310	311	312	313	314	315	316	317	318	319	320	321	322	323	324	325	326	327	328	329	330	331	332	333	334	335	336	337	338	339	340	341	342	343	344	345	346	347	348	349	350	351	352	353	354	355	356	357	358	359	360	361	362	363	364	365	366	367	368	369	370	371	372	373	374	375	376	377	378	379	380	381	382	383	384	385	386	387	388	389	390	391	392	393	394	395	396	397	398	399	400	401	402	403	404	405	406	407	408	409	410	411	412	413	414	415	416	417	418	419	420	421	422	423	424	425	426	427	428	429	430	431	432	433	434	435	436	437	438	439	440	441	442	443	444	445	446	447	448	449	450	451	452	453	454	455	456	457	458	459	460	461	462	463	464	465	466	467	468	469	470	471	472	473	474	475	476	477	478	479	480	481	482	483	484	485	486	487	488	489	490	491	492	493	494	495	496	497	498	499	500	501	502	503	504	505	506	507	508	509	510	511	512	513	514	515	516	517	518	519	520	521	522	523	524	525	526	527	528	529	530	531	532	533	534	535	536	537	538	539	540	541	542	543	544	545	546	547	548	549	550	551	552	553	554	555	556	557	558	559	560	561	562	563	564	565	566	567	568	569	570	571	572	573	574	575	576	577	578	579	580	581	582	583	584	585	586	587	588	589	590	591	592	593	594	595	596	597	598	599	600	601	602	603	604	605	606	607	608	609	610	611	612	613	614	615	616	617	618	619	620	621	622	623	624	625	626	627	628	629	630	631	632	633	634	635	636	637	638	639	640	641	642	643	644	645	646	647	648	649	650	651	652	653	654	655	656	657	658	659	660	661	662	663	664	665	666	667	668	669	670	671	672	673	674	675	676	677	678	679	680	681	682	683	684	685	686	687	688	689	690	691	692	693	694	695	696	697	698	699	700	701	702	703	704	705	706	707	708	709	710	711	712	713	714	715	716	717	718	719	720	721	722	723	724	725	726	727	728	729	730	731	732	733	734	735	736	737	738	739	740	741	742	743	744	745	746	747	748	749	750	751	752	753	754	755	756	757	758	759	760	761	762	763	764	765	766	767	768	769	770	771	772	773	774	775	776	777	778	779	780	781	782	783	784	785	786	787	788	789	790	791	792	793	794	795	796	797	798	799	800	801	802	803	804	805	806	807	808	809	810	811	812	813	814	815	816	817	818	819	820	821	822	823	824	825	826	827	828	829	830	831	832	833	834	835	836	837	838	839	840	841	842	843	844	845	846	847	848	849	850	851	852	853	854	855	856	857	858	859	860	861	862	863	864	865	866	867	868	869	870	871	872	873	874	875	876	877	878	879	880	881	882	883	884	885	886	887	888	889	890	891	892	893	894	895	896	897	898	899	900	901	902	903	904	905	906	907	908	909	910	911	912	913	914	915	916	917	918	919	920	921	922	923	924	925	926	927	928	929	930	931	932	933	934	935	936	937	938	939	940	941	942	943	944	945	946	947	948	949	950	951	952	953	954	955	956	957	958	959	960	961	962	963	964	965	966	967	968	969	970	971	972	973	974	975	976	977	978	979	980	981	982	983	984	985	986	987	988	989	990	991	992	993	994	995	996	997	998	999	1000
---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	------

7. Considere que o segmento de dados de um programa contém as seguintes diretivas:
- ```

L1: .word 0, 0, 4, 4
L2: .ascii "ExameJan23"
    .align 3
L3: .space 16
L4:

```
- Handwritten notes: 40 / 34 / 30 / 38, 14, 16, 28, 10010A10, 10010A10, 16 -> 436

Se o endereço correspondente ao label L1 for  $0x10010A10$ , o endereço a que corresponde o label L4 é:

- a.  $0x10010A40$  b.  $0x10010A34$  c.  $0x10010A30$  d.  $0x10010A28$

8. Considere um sistema computacional em que existe uma memória do tipo ROM (Read Only) que armazena programas, e uma memória do tipo RAM que armazena dados. Sabendo que as duas memórias estão em espaços de endereçamento distintos, pode afirmar-se que estamos perante uma arquitetura:

- a. do tipo Harvard.  
b. mista Harvard/von Neumann.  
c. do tipo von Neumann.  
d. do tipo von Neumann com armazenamento não volátil de programas.

Considere o seguinte segmento de código assembly:

```

lui    $3, 0xC614
ori    $4, $3, 0x19A5
sw     $4, 0x40($6)
lbu    $8, 0x41($6)

```

Após a execução do código anterior, o valor de \$8, num MIPS little endian, é:

- a.  $0x000000C6$  b.  $0x00000019$  c.  $0x00000014$  d.  $0x0000004C$

O modo de endereçamento utilizado na instrução `lbu $t0, 4($ra)` é:

a. por registo com deslocamento.

b. por registo.

c. ao PC.

d. direto.

Uma instrução `jal sqblz`, armazenada no endereço de memória  $0x004004CC$ . O label `jal` está no endereço  $0x00400518$ . Na conclusão da execução da instrução, o registo `$ra` contém:

a. o valor  $0x00400518$ .

b. o valor  $0x004004D0$ .

c. o valor  $0x0040051C$ .

d. o valor  $0x004004CC$ .

Exame

Arquitetura de Computadores I

2020

7. Considere que o segmento de dados de um programa contém as seguintes diretivas:

```
L1: word 0, 0  
L2: ascii "Exame-Jan23"  
align 3  
L3: space 16  
L4:
```

Se o endereço correspondente ao label L1 for 0x10010A10, o endereço a que corresponde o label L4 é:

- a. 0x10010A40      b. 0x10010A34      c. 0x10010A30      d. 0x10010A38

8. Considere um sistema computacional em que existe uma memória do tipo ROM (Read Only Memory), que armazena programas, e uma memória do tipo RAM que armazena dados. Sabendo que as duas memórias residem em espaços de endereçamento distintos, pode afirmar-se que estamos perante uma arquitetura:
- a. do tipo Harvard.  
b. mista Harvard/von Neumann.  
c. do tipo von Neumann.  
d. do tipo von Neumann com armazenamento não volátil de programas.

9. Considere o seguinte segmento de código assembly:

```
lui $3, 0xC6  
ori $4, $3, 0x19  
sw $4, 0x40($6)  
lbu ($8), 0x41($6)
```

No final da execução do código anterior, o valor de \$8, num MIPS little endian

- a. 0x000000C6      b. 0x00000019      c. 0x00000014      d. 0x000000C6

o endereço de endereçamento utilizado na instrução lb \$t0, 4(\$ra) é:

Considera-se um sistema computacional em que existe uma memória principal, uma memória cache e uma memória de backup. A memória principal é de tipo RAM e a memória de backup é de tipo ROM. A memória cache é de tipo RAM e a memória de backup é de tipo ROM. A memória principal e a memória de backup são de tipo RAM e a memória cache é de tipo ROM.

8. Considere um sistema computacional em que existe uma memória principal, uma memória cache e uma memória de backup. A memória principal é de tipo RAM e a memória de backup é de tipo ROM. A memória cache é de tipo RAM e a memória de backup é de tipo ROM. A memória principal e a memória de backup são de tipo RAM e a memória cache é de tipo ROM.

9. Considere o seguinte segmento de código assembly:

```
lui    $3, 0xc614
ori    $4, $3, 0x18A5
sw     $4, 0x40($6)
lb     $8, 0x41($6)
```

No final da execução do código anterior, o valor de \$8, num MIPS little endian, é:

- a. 0x000000c6      b. 0x00000019      c. 0x00000014      d. 0x000000A5

10. O principal modo de endereçamento utilizado na instrução `lb $t0, 4($ra)` é:

- a. indireto por registo com deslocamento.  
b. indireto por registo.  
c. relativo ao PC.  
d. pseudo-direto.



11. Considere a instrução `jal sqblz`, armazenada no endereço de memória `0x004004cc`. O label `ao` ao endereço `0x00400518`. Na conclusão da execução da instrução, o registo `$ra`:

- a. terá armazenado o valor `0x00400518`.  
b. terá armazenado o valor `0x004004D0`.  
c. terá armazenado o valor `0x0040051C`.  
d. terá armazenado o valor `0x004004CC`.





11. Considere o código de instruções utilizado na instrução `addi $t0, $t0, 5`:
- a. endereço por deslocamento deslocado
  - b. endereço por deslocamento
  - c. endereço por deslocamento
  - d. endereço por deslocamento
12. Considere a instrução `addi $t0, $t0, 5`, armazenada no endereço de memória hexadecimal `0x00400000`. O valor `5` corresponde ao endereço `0x00400005`. Na conclusão da execução da instrução, o registro `$t0`:
- a. terá armazenado o valor `0x00400005`.
  - b. terá armazenado o valor `0x00400000`.
  - c. terá armazenado o valor `0x00400001`.
  - d. terá armazenado o valor `0x00400006`.
13. Se pretender que, num programa de MIPS, o fluxo de execução seja desviado de forma incondicional para qual endereço múltiplo de quatro do espaço de endereçamento da CPU, deverá utilizar a instrução do tipo:
- a. `j label`
  - b. `jr $reg`
  - c. `jal label`
  - d. `beq $0, $0, label`



13. Na implementação *multi-cycle* de um processador MIPS, durante a execução da instrução `xor` a unidade de controlo ativa o sinal:
- a. `RegWrite` no quarto ciclo de relógio.
  - b. `IRWrite` no segundo ciclo de relógio.
  - c. `MemWrite` no primeiro ciclo de relógio.
  - d. `MemRead` no quarto ciclo de relógio.



Cotações: Grupo I: cada 0.6 valores; Grupo II: cada 0.8 valores.

14. A detecção de overflow numa operação de adição de números inteiros sem sinal faz-se através:
- da avaliação do bit mais significativo do resultado.
  - do xor entre o carry in e o carry out da célula de 1 bit mais significativa do resultado.
  - do xor entre os 2 bits mais significativos do resultado.
  - da avaliação do carry out do bit mais significativo do resultado.

15. Considerando que \$1=0xFFFFFFFF e \$4=0xFFFFFFFF, o conteúdo dos registos HI e LO após a execução da instrução mult \$1, \$4 (multiplicação signed) é:
- HI=0x00000000, LO=0x00000012
  - HI=0xFFFFFFFF, LO=0x00000018
  - HI=0xFFFFFFFF, LO=0xFFFFFFFF
  - HI=0x00000000, LO=0xFFFFFFFF

16. Em linguagem C, o código que permite atribuir o valor 0x1F ao elemento de índice 5 do array "t" é:

|                           |                        |                          |                        |
|---------------------------|------------------------|--------------------------|------------------------|
| a.                        | b.                     | c.                       | d. ?                   |
| int t[20];<br>int *mp;    | int t[20];<br>int *mp; | int t[20];<br>int *mp;   | int t[20];<br>int *mp; |
| mp=t[0];<br>*(mp+5)=0x1F; | mp=t;<br>*(mp+5)=0x1F; | mp=&t;<br>*(mp+20)=0x1F; | mp=t;<br>*(mp)+5=0x1F; |

17. Numa implementação *single-cycle* da arquitetura MIPS, semelhante à apresentada nas aulas, na transição de sinal de relógio:

- é escrito o resultado produzido pela instrução em execução no elemento de estado respetivo à execução da instrução seguinte.
- é armazenado no PC o resultado da operação realizada na ALU para cálculo de PC+4.
- é realizada a leitura síncrona da memória de instruções.
- é realizada a leitura assíncrona do banco de registos.

Quando uma operação é realizada, a representação de um processador ocorre quando  
 A. os dados necessários para a execução de uma instrução não se encontram alinhados e o operando  
 B. os dados necessários para a execução de uma instrução não se encontram alinhados e o operando  
 C. os dados necessários para a execução de uma instrução não se encontram alinhados e o operando  
 D. os dados necessários para a execução de uma instrução não se encontram alinhados e o operando

22. A representação de quantidade  $-787,5 \times 10^{-2}$  no formato IEEE 754 precisão simples, é:
- a. 0xc0e40000
  - b. 0x8f6e0000
  - c. 0x30f80000
  - d. 0xc0f80000

Grupo II

$$-787,5 \times 10^{-2} \text{ no formato IEEE 754 precisão simples, é:}$$

$$= -7,875 \times 10^0$$

$$= -7,875 \text{ base 10}$$

23. Qual a instrução virtual do MIPS que é implementada pela seguinte sequência de instruções nativas:
- a. ori \$1, \$0, 0x3C
  - b. sll \$1, \$1, \$8
  - c. beq \$1, \$0, L1
  - d. bge \$8, 0x3C, L1
  - e. bgt \$8, 0x3C, L1
  - f. blt \$8, 0x3C, L1
  - g. \$8, 0x3C, L1

igual

cc1  
 11 00 0000 1000

$$128 + 1 = 129 - 127 = -2$$

3F48 = 0011 1111 1111 1111

3F3C = 0011 1111 1111 1100

0xc0e40000 e \$f6=0xc0800000

27. Na situação apresentada na questão anterior, admissa que no instante zero, imediatamente a  
 X sinal de relógio, vai iniciar-se a instrução *flush* da primeira instrução. O valor a saída da ALU no oitavo ciclo de relógio, contando a partir do instante zero, é:
- 0x00000000
  - 0x0000000C
  - 0x00000001
  - 0x003B863D

28. Numa implementação *pipelined* da arquitetura MIPS, o hazard de dados existente na sequência de in-  
 X *lw \$1, 0(\$2)* seguida de *beq \$1, \$3, target*, pode ser resolvido:
- com *stall* durante 2 ciclos de relógio.
  - sem recurso a *stalling*, com *forwarding* de EX/MEM para EX.
  - com *stall* durante 1 ciclo de relógio seguido de *forwarding* de MEM/WB para EX.
  - com *stall* durante 1 ciclo de relógio seguido de *forwarding* de EX/MEM para ID.



29. O código máquina da instrução *lw \$15, -4(\$3)*, representado em hexadecimal, é (considerando  
 instrução, *opcode* = 0x23):

- 0x8DE3FFFC
- 0x8C83FFF8
- 0x8C6FFFFC
- 0x8DE3FFF8



~~0x8DE3FFFC~~  
 0x23 | \$15 | \$3 + 4

Grupo I: cada 0.6 valores; Grupo II: cada 0.8 valores.



15. Considere-se que  $R1=0xFFFFFFFF$  e  $R2=0xFFFFFFFF$ , o conteúdo dos registos  $R1$  e  $R2$  após a execução da instrução `mult R1, R2` (multiplicação signed) é:
- $R1=0x00000000$ ,  $R2=0x00000000$
  - $R1=0xFFFFFFFF$ ,  $R2=0x00000000$
  - $R1=0xFFFFFFFF$ ,  $R2=0xFFFFFFFF$
  - $R1=0x00000000$ ,  $R2=0xFFFFFFFF$

1111... 10  
1111... 1101

16. Em linguagem C, o código que permite atribuir o valor `0x1F` ao elemento de índice 5 do array "t" é:

- |                           |                           |                            |                            |
|---------------------------|---------------------------|----------------------------|----------------------------|
| a.                        | b.                        | c.                         | d.                         |
| <code>t[5] = 0x1F;</code> | <code>t[5] = 0x1F;</code> | <code>t[20] = 0x1F;</code> | <code>t[20] = 0x1F;</code> |
| <code>*mp = 0x1F;</code>  | <code>*mp = 0x1F;</code>  | <code>*mp = 0x1F;</code>   | <code>*mp = 0x1F;</code>   |
|                           |                           | <code>*mp = 0x1F;</code>   | <code>*mp = 0x1F;</code>   |

17. Na implementação *single-cycle* da arquitetura MIPS, semelhante à apresentada nas aulas, na transição ativa do sinal de relógio:
- é escrito o resultado produzido pela instrução em execução no elemento de estado respectivo e inicia-se a execução da instrução seguinte.
  - é armazenado no PC o resultado da operação realizada na ALU para cálculo de  $PC+4$ .
  - é realizada a leitura síncrona da memória de instruções.
  - é realizada a leitura assíncrona do banco de registos.

A unidade de *forwarding* de uma implementação *pipelined* da arquitetura MIPS é um bloco: combinatório responsável pelo avanço das instruções no pipeline. combinatório responsável pela geração dos sinais de controlo que permitem o encaminhamento de um res para estágios mais avançados do pipeline. combinatório que deteta a dependência entre instruções que se encontram em execução no pipeline e q sinais de controlo que permitem a utilização de resultados produzidos por instruções que se encont estágios de execução mais avançados. ado numa máquina de estados que deteta situações em que uma dada instrução pode avançar, num o, mais do que um estágio no pipeline.

O ciclo de relógio de execução da instrução *SW* numa arquitetura MIPS *multi-cycle* é calculado em função do conteúdo do:   
 - campo *RT* da instrução com o valor do *offset sign extended* para 32 bits.   
 - campo *RS* da instrução com o valor do *offset* estendido com zeros para 32 bits.   
 - campo *RS* da instrução com o valor do *offset sign extended* para 32 bits.   
 - valor multiplicado por 4 do *offset sign extended* para 32 bits.

Na notação IEEE 754, precisão dupla, um expoente positivo é codificado:   
 - superior a 1023 e igual ou inferior a 2046.   
 - inferior a 0 e inferior a 1024.   
 - inferior a 128 e inferior a 255.   
 - o bit mais significativo a 0.

1024 -  
e - 1023



$$-7,875 \times 10^{-2} = -7,875 = -111.111 \times 2^0 = -1.1111 \times 2^2$$

$$E = 2 + 127 = 129$$

|            |
|------------|
| 0,875      |
| $\times 2$ |
| 1,750      |
| $\times 2$ |
| 1,500      |
| $\times 2$ |
| 1,000      |

Parte Decimal

| SINAL | Expoente  | Mantissa (23 bits) |
|-------|-----------|--------------------|
| 1     | 100,00001 | 111,1100, (0)      |
|       | C 0 F C   |                    |