

ĐẠI HỌC QUỐC GIA TP.HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA KHOA ĐIỆN – ĐIỆN TỬ

BỘ MÔN ĐIỆN TỬ

.....o0o.....



BÁO CÁO

BÀI TẬP LỚN CẤU TRÚC MÁY TÍNH

Đề tài:

**Thiết kế CPU chạy đơn chu kỳ và Pipeline
dựa trên cấu trúc CPU RV32**

GVHD:Thầy Trần Hoàng Linh

SVTT:Dương Nhật Huy – 181016

1. Thiết kế CPU RV32

a) Phân tích

CPU RV32 có tổng cộng 32 lệnh hợp ngữ, trong đó mỗi lệnh có độ dài 32bit và có 7 bit [6:0] (opcode) để xác định loại lệnh.

imm[31:12]					rd	0110111	LUI		
imm[31:12]					rd	0010111	AUIPC		
imm[20:10:1 11 19:12]					rd	1101111	JAL		
imm[11:0]					rs1	000	rd	1100111	JALR
imm[12:10:5]		rs2	rs1	000	imm[4:1 11]		1100011	BEQ	
imm[12:10:5]		rs2	rs1	001	imm[4:1 11]		1100011	BNE	
imm[12:10:5]		rs2	rs1	100	imm[4:1 11]		1100011	BLT	
imm[12:10:5]		rs2	rs1	101	imm[4:1 11]		1100011	BGE	
imm[12:10:5]		rs2	rs1	110	imm[4:1 11]		1100011	BLTU	
imm[12:10:5]		rs2	rs1	111	imm[4:1 11]		1100011	BGEU	
imm[11:0]					rs1	000	rd	0000011	LB
imm[11:0]					rs1	001	rd	0000011	LH
imm[11:0]					rs1	010	rd	0000011	LW
imm[11:0]					rs1	100	rd	0000011	LBU
imm[11:0]					rs1	101	rd	0000011	LHU
imm[11:5]		rs2	rs1	000	imm[4:0]		0100011	SB	
imm[11:5]		rs2	rs1	001	imm[4:0]		0100011	SH	
imm[11:5]		rs2	rs1	010	imm[4:0]		0100011	SW	
imm[11:0]					rs1	000	rd	0010011	ADDI
imm[11:0]					rs1	010	rd	0010011	SLTI
imm[11:0]					rs1	011	rd	0010011	SLTIU
imm[11:0]					rs1	100	rd	0010011	XORI
imm[11:0]					rs1	110	rd	0010011	ORI
imm[11:0]					rs1	111	rd	0010011	ANDI
0000000		shamt	rs1	001	rd		0010011	SLLI	
0000000		shamt	rs1	101	rd		0010011	SRLI	
0100000		shamt	rs1	101	rd		0010011	SRAI	
0000000		rs2	rs1	000	rd		0110011	ADD	
0100000		rs2	rs1	000	rd		0110011	SUB	
0000000		rs2	rs1	001	rd		0110011	SLL	
0000000		rs2	rs1	010	rd		0110011	SLT	
0000000		rs2	rs1	011	rd		0110011	SLTU	
0000000		rs2	rs1	100	rd		0110011	XOR	
0000000		rs2	rs1	101	rd		0110011	SRL	
0100000		rs2	rs1	101	rd		0110011	SRA	
0000000		rs2	rs1	110	rd		0110011	OR	
0000000		rs2	rs1	111	rd		0110011	AND	

- Tập lệnh của RV32 còn được gọi là tập lệnh kiểu load-store, điều đó có nghĩa là data trong bộ nhớ muốn được thực thi thì trước hết phải được lấy ra bỏ vào băng thanh ghi rồi mới được tính toán. Sau khi tính toán, data sẽ được lưu lại vào memory.

- Các thanh ghi trong băng thanh ghi (Register Bank) có độ dài 32 bits và có 32 thanh ghi (từ x0 – x31) \Rightarrow Cần có 5 bits để xác định địa chỉ của các thanh ghi trong băng thanh ghi. Trong đó, chức năng của 32 thanh ghi được cho như ở bảng dưới.

Register	ABI Name	Description	Saver
x0	zero	Hard-wired zero	—
x1	ra	Return address	Caller
x2	sp	Stack pointer	Callee
x3	gp	Global pointer	—
x4	tp	Thread pointer	—
x5	t0	Temporary/alternate link register	Caller
x6–7	t1–2	Temporaries	Caller
x8	s0/fp	Saved register/frame pointer	Callee
x9	s1	Saved register	Callee
x10–11	a0–1	Function arguments/return values	Caller
x12–17	a2–7	Function arguments	Caller
x18–27	s2–11	Saved registers	Callee
x28–31	t3–6	Temporaries	Caller

Thanh ghi x0 luôn có giá trị bằng 0x00000000 và không thay đổi giá trị này.

- Nhóm lệnh R-Format:

Nhóm lệnh này bao gồm các lệnh có cấu trúc như ở hình sau:

0000000	rs2	rs1	000	rd	0110011	ADD
0100000	rs2	rs1	000	rd	0110011	SUB
0000000	rs2	rs1	001	rd	0110011	SLL
0000000	rs2	rs1	010	rd	0110011	SLT
0000000	rs2	rs1	011	rd	0110011	SLTU
0000000	rs2	rs1	100	rd	0110011	XOR
0000000	rs2	rs1	101	rd	0110011	SRL
0100000	rs2	rs1	101	rd	0110011	SRA
0000000	rs2	rs1	110	rd	0110011	OR
0000000	rs2	rs1	111	rd	0110011	AND

Nhóm lệnh này có opcode là [6:0] = 0110011

Nhóm lệnh này thực hiện lấy hai giá trị lưu ở thanh ghi rs1 và rs2 thực hiện đưa vào khối ALU để tính toán, sau đó lưu kết quả vào thanh ghi rd.

- Nhóm lệnh I (Tính toán):

imm[11:0]		rs1	000	rd	0010011	addi
imm[11:0]		rs1	010	rd	0010011	slti
imm[11:0]		rs1	011	rd	0010011	sltiu
imm[11:0]		rs1	100	rd	0010011	xori
imm[11:0]		rs1	110	rd	0010011	ori
imm[11:0]		rs1	111	rd	0010011	andi
0000000	shamt	rs1	001	rd	0010011	slli
0000000	shamt	rs1	101	rd	0010011	srli
0100000	shamt	rs1	101	rd	0010011	srai

Nhóm lệnh này có opcode là [6:0] = 0010011

Nhóm lệnh này (trừ 3 lệnh SRAI, SRLI, SLLI) thực hiện lấy giá trị lưu ở thanh ghi rs1 và giá trị lưu ở imm[11:0] (được mở rộng dấu), thực hiện đưa vào khối ALU để tính toán. Kết quả được lưu vào thanh ghi rd.

- Nhóm lệnh I (Load data)

imm[11:0]	rs1	000	rd	0000011	lb
imm[11:0]	rs1	010	rd	0000011	lh
imm[11:0]	rs1	011	rd	0000011	lw
imm[11:0]	rs1	100	rd	0000011	lbu
imm[11:0]	rs1	110	rd	0000011	lhu

Nhóm lệnh này có opcode là [6:0] = 0000011

Nhóm lệnh này thực hiện lấy hai giá trị lưu ở thanh ghi rs1 và giá trị lưu ở imm[11:0] (được mở rộng dấu) để tính tổng rs1 + ext(imm[11:0]). Sau đó lấy giá trị lưu trong data memory tại địa chỉ rs1 + ext(imm[11:0]). Với lb, lh, lw lần lượt lấy 8, 16, 32 bit của giá trị này lưu vào rd. Do lb và lh chỉ lấy 8, 16 bit nên các bit trống của rd được lấp đầy bằng dấu của giá trị lấy ra từ data memory

- Nhóm lệnh S (Store data)

Imm[11:5]	rs2	rs1	000	imm[4:0]	0100011	sb
Imm[11:5]	rs2	rs1	001	imm[4:0]	0100011	sh
Imm[11:5]	rs2	rs1	010	imm[4:0]	0100011	sw

Nhóm lệnh này có opcode là [6:0] = 0100011

Nhóm lệnh này thực hiện lấy hai giá trị lưu ở thanh ghi rs1 và giá trị lưu ở imm[11:5] và imm[4:0] (ghép lại và mở rộng dấu) để tính tổng rs1 + ext(imm[11:5]imm[4:0]). Sau đó lấy giá trị lưu trong thanh ghi rs2. Với các lệnh sb, sh, sw lưu 8, 16, 32 bit thấp của giá trị này vào DMEM tại địa chỉ rs1 + ext(imm[11:5]imm[4:0]).

- Nhóm lệnh B (rẽ nhánh)

imm[12 10:5]	rs2	rs1	000	imm[4:1 11]	1100011	BEQ
imm[12 10:5]	rs2	rs1	001	imm[4:1 11]	1100011	BNE
imm[12 10:5]	rs2	rs1	100	imm[4:1 11]	1100011	BLT
imm[12 10:5]	rs2	rs1	101	imm[4:1 11]	1100011	BGE
imm[12 10:5]	rs2	rs1	110	imm[4:1 11]	1100011	BLTU
imm[12 10:5]	rs2	rs1	111	imm[4:1 11]	1100011	BGEU

Nhóm lệnh này có opcode là [6:0] = 1100011

Nhóm lệnh này sẽ thực hiện chuyển giá trị của thanh ghi PC thành giá trị được lưu trong các phần imm giá trị lưu trong rs1 và rs2 thỏa điều kiện câu lệnh (bằng, không bằng, nhỏ hơn, lớn hơn, nhỏ hơn không dấu, lớn hơn không dấu)

Khi lấy giá trị lưu ở phần imm ta phải ghép lại cho đúng thứ tự và mở rộng dấu, bit LSB luôn luôn bằng 0.

- Nhóm lệnh U

31	12	11	7	6	0
imm[31:12]			rd	opcode	

+ Với lệnh LUI :

Opcode = 0110111. Lệnh này load giá trị imm[31:12]000000000000 vào thanh ghi rd.

+ Với lệnh AUIPC

Opcode = 0010111. Lệnh này load giá trị ở PC vào thanh ghi rd.

- Nhóm lệnh J (nhảy không điều kiện)

imm[20 10:1 11 19:12]				rd	1101111	JAL
imm[11:0]		rs1	000	rd	1100111	JALR

+ Với lệnh JAL :

$x[rd] = pc+4; pc = pc + ext(imm);$

+ Với lệnh JALR:

$x[rd] = pc+4; pc = (x[rs1]+ext(imm)) \& \sim 1;$

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																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																								
								=> PC=PC+4	000 => Inst[31:20], ext, sign								0 => ReadOnly								0 => sign								0 => B								0 => A								0000 => Add								0 => Read								00 => W								00 => DMEM																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																						
								1 => PC from ALU	001 => Inst[31:20], ext, sign								1 => Write								1 => usign								1 => imm								1 => PC								0001 => Sub								01 => B								001 => W -> extB								01 => ALU																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																						
								010 => Inst[24:20], ext, usign																																0010 => Shift, left								1 => Write								10 => HW								010 => W -> extHW								10 => PC +4																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																							
								011 => Inst[25][11:7], ext, sign																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																							

37	25	S	SB	x	000	01000	x	x	0	011	0	x	1	0	0000	1	01	x	x
38	26	S	SH	x	001	01000	x	x	0	011	0	x	1	0	0000	1	10	x	x
39	27	S	SW	x	010	01000	x	x	0	011	0	x	1	0	0000	1	00	x	x
40																			
41	28	B	BEQ	x	000	11000	1	x	1	100	0	x	1	1	0000	0	x	x	x
42							0		0										
43	29	B	BNE	x	001	11000	0	x	1	100	0	x	1	1	0000	0	x	x	x
44							0		0										
45	30	B	BLT	x	100	11000	x	1	1	100	0	0	1	1	0000	0	x	x	x
46							0		0										
47	31	B	BGE	x	101	11000	x	0	1	100	0	0	1	1	0000	0	x	x	x
48							1		0										
49	32	B	BLTU	x	110	11000	x	1	1	100	0	1	1	1	0000	0	x	x	x
50							0		0										
51	33	B	BGEU	x	111	11000	x	0	1	100	0	1	1	1	0000	0	x	x	x
52							1		0										
53																			
54	34	U	LUI	x	x	01101	x	x	0	101	1	x	1	x	1010	0	x	x	01
55	35	U	AUIPC	x	x	00101	x	x	0	101	1	x	1	1	0000	0	x	x	01
56																			
57	36	J	JAL	x	x	11011	x	x	1	110	1	x	1	1	0000	0	x	x	10
58	37	J	JALR	x	x	11001	x	x	1	000	1	x	1	0	0000	0	x	x	10

c) Kiểm tra thiết kế

Sử dụng phần mềm thực hiện các lệnh trong risc – v.

```
lui x1, 983041
addi x1, x1, 31
addi x3, x3, 8
sw x1, 8(x0)
lb x2, 0(x3)
lbu x4, 0(x3)
blt x1, x2, LABEL
addi x4, x3, 10
LABEL: addi x4, x4, 32
```

Chuyển đổi các câu lệnh sang mã thập lục phân và lưu vào file instdata.mem

instmem.data -

File Edit Format

```
f00010b7
01f08093
00818193
00102423
00018103
0001c203
0020c463
00a18213
02020213
```

ck_j	1h1																		
rst_r	1h0																		
nxt_pc_o	32h...	32h00000004				32h00000008	32h0000000c	32h00000010	32h00000014	32h00000018	32h0000001c	32h00000020	32h00000024						
pc_o	32h...	32h00000000				32h00000004	32h00000008	32h0000000c	32h00000010	32h00000014	32h00000018	32h0000001c	32h00000020	32h00000024					
pc_four_o	32h...	32h00000004				32h00000008	32h0000000c	32h00000010	32h00000014	32h00000018	32h0000001c	32h00000020	32h00000024						
rs1_data_o	32h...	32h00000000				32h00001000	32h00000000	32h00000000	32h00000008	32h00000010	32h0000001f	32h00000000	32h00000000						
rs2_data_o	32h...	32h00000000				32h00001000	32h00000000	32h0000101f	32h00000000	32h00000000	32h0000001f	32h00000000	32h00000000						
instr_o	32h...	32h000010b7				32h01f08093	32h00818193	32h00102423	32h00018103	32h0001c203	32h0020c463	32h0020213							
imm_o	32h...	32h000010b7				32h0000000f	32h00000008	32h00000000	32h00000000	32h00000008	32h00000008	32h00000020	32h00000020						
akumux1_o	32h...	32h00000000				32h00001000	32h00000000	32h00000000	32h00000008	32h00000008	32h00000018	32h0000001f	32h00000000						
akumux2_o	32h...	32h000010b7				32h0000000f	32h00000008	32h00000000	32h00000000	32h00000008	32h00000008	32h00000020	32h00000020						
ak_data_o	32h...	32h000010b7				32h0000101f	32h00000008	32h00000008	32h00000008	32h00000008	32h00000020	32h00000020	32h00000020						
ld_data_o	32h...	32h000010b7				32h0000101f	32h00000008	32h00000008	32h00000008	32h00000008	32h00000020	32h00000020	32h00000020						
wb_data_o	32h...	32h000010b7				32h0000101f	32h00000008	32h00000008	32h00000008	32h00000008	32h00000020	32h00000020	32h00000020						
br_less_o	1h0																		
br_equal_o	1h0																		
br_sel_o	1h0																		
op_a_sel_o	1h0																		
op_b_sel_o	1h0																		
rd_vren_o	1h0																		
mem_vren_o	1h0																		
br_unsigned_o	1hx																		
imm_sel_o	3h0					3h0		3h1		3h0			3h2		3h0				
alu_op_o	4h0					4h0													
wb_sel_o	2h1					2h1				2h0					2h1				
ctrl_o	18h...	18h000009				18h000001		18h10000x		18h000000			18h000018x		18h000001				
take_bit_o	3hx					3hx		3h2		3h0									

- Câu lệnh 1:
 - + pc = 0
 - + x1 = f0001000h
- Câu lệnh 2:
 - + pc = 4h

+ x1 = x1 + 31 = f000101fh

- Câu lệnh 3:

+ pc = 8h

+ x3 = x3 + 8 = 0 + 8 = 8

- Câu lệnh 4:

+ pc = ch

+ Lưu giá trị x1 = f000101fh vào địa chỉ x0 + 8h = 8h trong datamem

Câu lệnh 5:

+ pc = 10h

+ Vào datamem tại địa chỉ x3 + 0 = 8h , lấy 8 bit lưu vào x2. Do dùng lệnh lb và giá trị f0001000h là một số âm nếu là số có dấu nên các bit trống x2 điền giá trị 1.

X2 = fffffff1h

Câu lệnh 6:

+ pc = 14h

+ Vào datamem tại địa chỉ x3 + 0 = 8h , lấy 8 bit lưu vào x2. Do dùng lệnh lbu nên các bit trống x4 điền giá trị 0 . x4 = 1fh

Câu lệnh 7:

+ pc = 18h

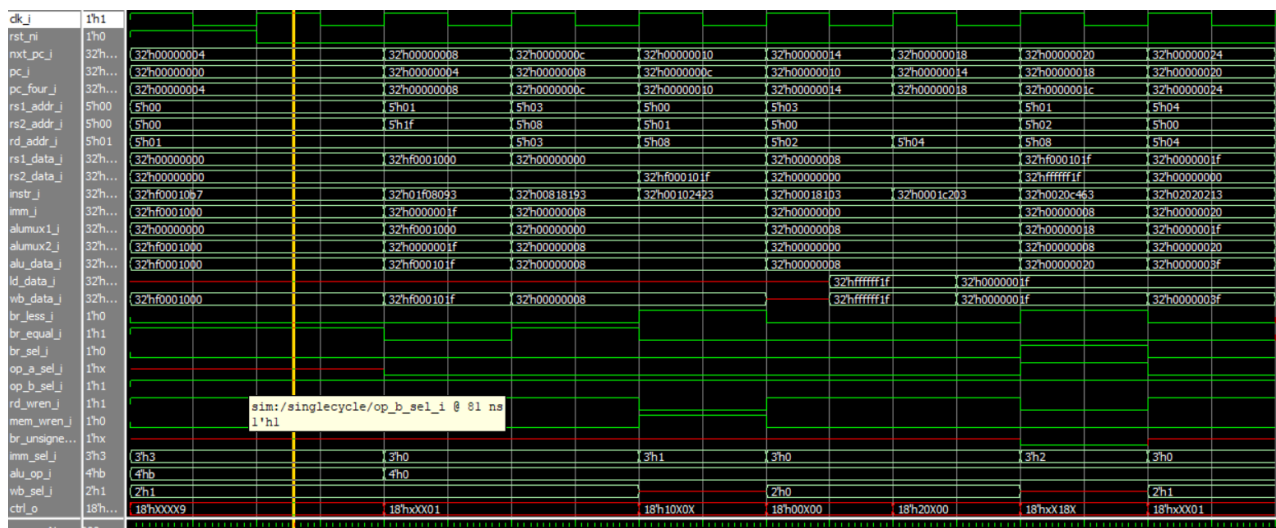
+ Nếu x1 < x2 (có dấu) thì nhảy đến chỗ LABEL . Do x1 = f000101f < fffffff1f = x2 nếu so sánh có dấu nên nhảy qua câu lệnh 8 đến câu lệnh 9 (pc = 20h)

Câu lệnh 9 :

+ pc = 20h

+ x4 = x4 + 32 = 3fh

d) Kết quả mô phỏng



File regfile.data

regfile.data

File Edit Forr

```
// memory c
// instance
// format=t
00000000
f000101f
ffffff1f
00000008
0000003f
```

File datamem.data

datamem.data -

File Edit Format

```
// memory data
// instance=:
// format=bin
0000
0000
0000
0000
0000
0000
0000
0000
0000
1111
0001
0000
0001
0000
0000
0000
1111
```

e) Kết luận

Thiết kế chạy đúng với bài test đặt ra

2. Câu hỏi

a. Write a code to find the largest number of the Fibonacci sequence less than 32'd500. You will put the ceiling number in register x10 and save the value you find in 0x400

```
addi x1 , x0 , 1
addi x2 , x0 , 0x0500
addi x4, x0, 1
```

LABEL:

```
add x1 , x1 , x4
add x4 , x1, x0
addi x10 , x10, 1
blt x1 , x2 , LABEL
```

```
sw x4 , 0x0400 (x0)
```

b. Write a code in which you read an 8-bit binary in the address 0x500, convert it into 3 decimal digits, and then write them into 0x420, 0x410, and 0x400, respectively. For example, the value in 0x500 is 0xEE, which is 238, so “2” will be in 0x420, “3” in 0x410, and “8” in 0x400

```
lbu x1 , 0x500(x0)
addi x1 , x1 , 0xEE
addi x2 , x0 , 0
addi x3 , x0 , 1
addi x4 , x0 , 100
```

```
TABLE0:
addi x2 , x2 , 100
addi x5 , x5 , 1
blt x2 , x1 , LABEL0
```

```
sub x5,x5,x3
sub x2,x2,x4
sub x1,x1,x2
addi x2, x0, 0
addi x4, x0,10
```

```
LABEL1:
addi x2 , x2 , 10
addi x6, x6, 1
blt x2 , x1 , LABEL1
sub x2,x2,x4
sub x6,x6,x3
sub x7,x1,x2
```

```
sb x5,0x420(x0)
sb x6,0x410(x0)
sb x7,0x400(x0)
```