Student ID :   Name :

Supplement Page 1 – Tear Off

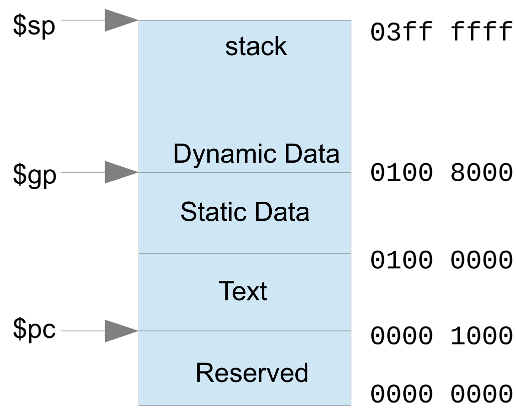
CS147DV Instruction Set

32-bit machine

32 internal registers

User addressable 0-27

28 is $gp or Global Pointer
29 is $sp or Stack Pointer
30 is $fp or Frame Pointer
31 is $ra or Return Pointer



Memory Allocation

Name	Mnemonic	Format	Operation	OpCode /funct
Addition	`add`	R	R[rd] = R[rs] + R[rt]	0x00 / 0x20
Subtraction	`sub`	R	R[rd] = R[rs] - R[rt]	0x00 / 0x22
Multiplication	`mul`	R	R[rd] = R[rs] * R[rt]	0x00 / 0x2c
Logical AND	`and`	R	R[rd] = R[rs] & R[rt]	0x00 / 0x24
Logical OR	`or`	R	R[rd] = R[rs] | R[rt]	0x00 / 0x25
Logical NOR	`nor`	R	R[rd] = ~(R[rs] | R[rt])	0x00 / 0x27
Set less than	`slt`	R	R[rd] = (R[rs] < R[rt])?1:0	0x00 / 0x2a
Shift left logical	`sll`	R	R[rd] = R[rs] << shamt	0x00 / 0x01
Shift right logical	`srl`	R	R[rd] = R[rs] >> shamt	0x00 / 0x02
Jump Register	`jr`	R	PC = R[rs]	0x00 / 0x08

Coding format: <mnemonic> <rd>, <rs>, <rt | shamt>

	opcode	rs	rt	rd	shamt	funct
R-type	opcode	rs	rt	rd	shamt	funct
	31 26	25 21	20 16	15 11	10 6	5 0

Student ID : Name :

Supplement Page 2 – Tear Off

CS147DV Instruction Set

Name	Mnemonic	Format	Operation	OpCode
Addition immediate	addi	I	R[rt] = R[rs] + SignExtImm	0x08
Multiplication immediate	muli	I	R[rt] = R[rs] * SignExtImm	0x1d
Logical AND immediate	andi	I	R[rt] = R[rs] & ZeroExtImm	0x0c
Logical OR immediate	ori	I	R[rt] = R[rs] | ZeroExtImm	0x0d
Load upper immediate	lui	I	R[rt] = {imm, 16'b0}	0x0f
Set less than immediate	slti	I	R[rt] = (R[rs] < SignExtImm)?1:0	0x0a
Branch on equal	beq	I	If (R[rs] == R[rt]) PC = PC + 1 + BranchAddress	0x04
Branch on not equal	bne	I	If (R[rs] != R[rt]) PC = PC + 1 + BranchAddress	0x05
Load word	lw	I	R[rt] = M[R[rs]+SignExtImm]	0x23
Store word	sw	I	M[R[rs]+SignExtImm] = R[rt]	0x2b

BranchAddress = {16{Imm[15]}, immediate }

Coding format: <mnemonic> <rt>, <rs>, <imm>

I-type	opcode	rs	rt	immediate
	31 26	25 21	20 16	15 0

13

Name	Mnemonic	Format	Operation	OpCode
Jump to address	jmp	J	PC = JumpAddress	0x02
Jump and Link	jal	J	R[31] = PC + 1; PC = JumpAddress	0x03
Push to Stack	push	J	M[$sp] = R[0] $sp = $sp - 1	0x1b
Pop from Stack	pop	J	$sp = $sp + 1 R[0] = M[$sp]	0x1c

JumpAddress = { 6'b0, address } // zero extend for 6 bit

Coding format: <mnemonic> <address>

J-type	opcode	address
	31 26	25 0

14

Student ID :   Name :

Supplement Page 3 – Tear Off

Size Units	
Unit	Value
T	2^{40}
G	2^{30}
M	2^{20}
K	2^{10}

Value Units	
Unit	Value
G	10^{9}
M	10^{6}
K	10^{3}
m	10^{-3}
u	10^{-6}
n	10^{-9}
p	10^{-12}
f	10^{-15}

Student ID :   Name :

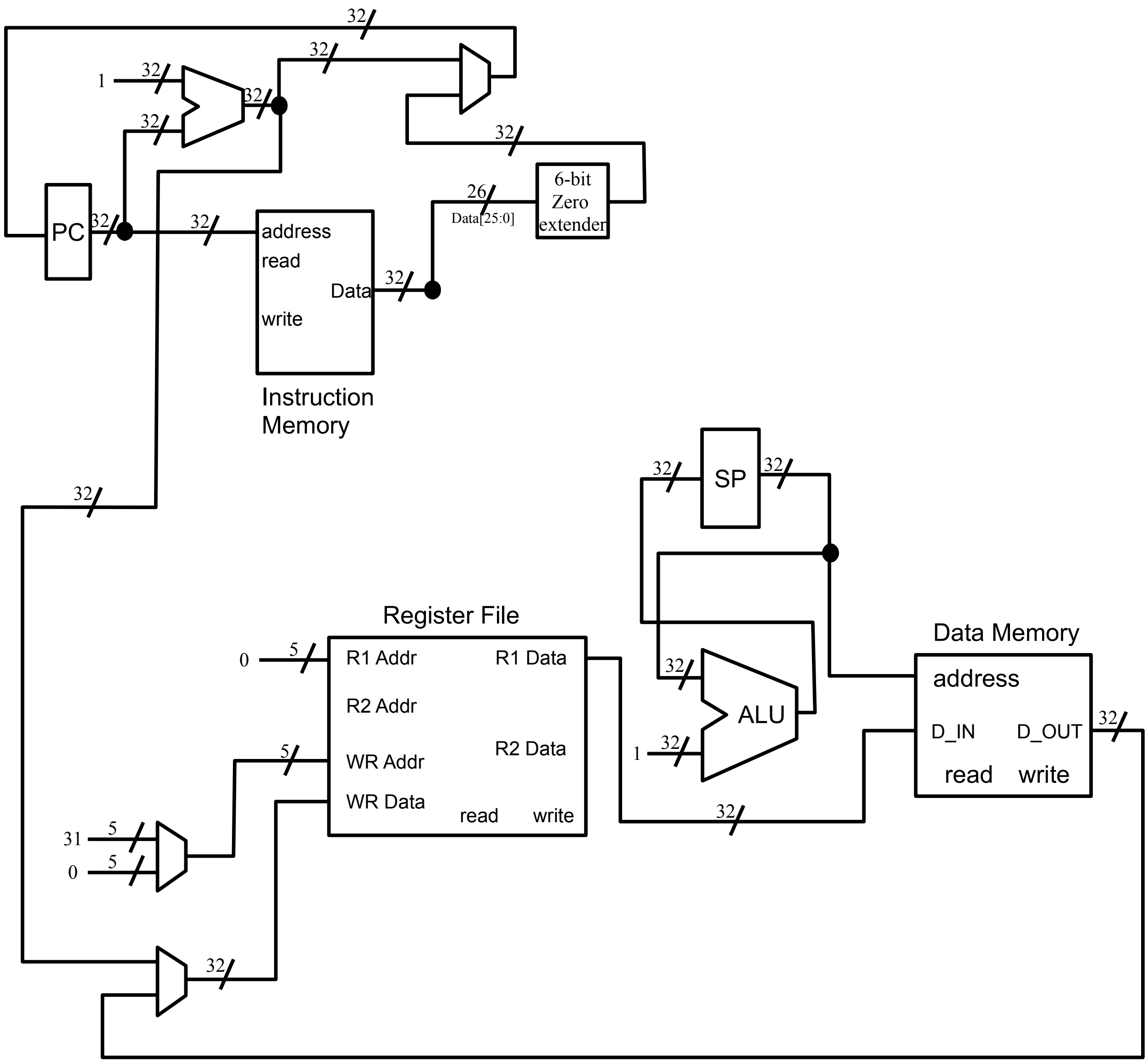
Class	CS147, Sec 01
Midterm	Spring 2017
Due Date	May 18, 2017 5:15 PM - 7:30 PM PST
Notes	1. Tear Off Supplemental pages 2. Closed book / note/ computer / internet exam 3. Paper pencil / pen exam – you may use calculator 4. Write your name and student ID at header section on **EVERY PAGE** 5. **Explanation** of answer is **required if applied**. 6. Q1- 4 are main questions to answer. 7. Q5 is extra credit question 8. Use back of the pages if needed to answer questions.

Please Don't turn page until instructed.

1. Regarding CS147DV ISA answer following question.
   (a) Draw complete data path supporting CS147DV J type instructions. [**1pts**]
   (b) What is power up address value for PC and SP in 32-bit hex format? [**1pts**]

ANS:

a)



b) PC = 0x00001000 , SP = 0x03FFFFFF

2. A 32-bit computing system uses byte addressable memory. If snippet of the memory at a time instance during program execution is like following diagram, answer data accessed (in hex format) for the following scenario. [No need to consider alignment issues]

   (a) Assuming the system using '**<u>Little-endian</u>**' assumption.
      i. Register Indirect Addressing with register content retrieved 0x00010007. [**0.5pts**]
      ii. Displacement Addressing with register content retrieved 0x0001000E and offset 0xFA (in 8-bit 2's complement format). [**0.5pts**]

   (b) Assuming the system using '**<u>Big-endian</u>**' assumption.
      i. Register Indirect Addressing with register content retrieved 0x00010002. [**0.5pts**]
      ii. Displacement Addressing with register content retrieved 0x00010002 and offset 0x0A (in 8-bit 2's complement format). [**0.5pts**]

Address	Data	Address	Data
0x00010012	0xA2	0x00010008	0x46
0x00010010	0x10	0x00010007	0x71
0x0001000F	0x17	0x00010006	0x56
0x0001000E	0x03	0x00010005	0x73
0x0001000D	0x06	0x00010004	0xA6
0x0001000C	0x10	0x00010003	0xC5
0x0001000B	0x80	0x00010002	0x64
0x0001000A	0xFB	0x00010001	0x68
0x00010009	0xC8	0x00010000	0x61

ANS : a) (i) M[0x00010007] = 0x0xFBC84671
(ii) 0xFA = -6_{10} => M[0x0001000E - 6_{10}] = M[0x00010008] = 0x80FBC846

b) (i) M[0x00010002] = 0x64C5A673
(ii) 0x0A = 10_{10} => M[0x00010002 + 10_{10}] = M[0x0001000C] = 0x10060317

3. A program runs in 10 sec on computer A with 4GHz clock frequency. A new computer B needs to be built to run the same program in 6 sec. It has been determined that the clock rate (F) can be increased a lot, but it will need the program to run 1.2 times of the number of clock cycle (N) that was required on computer A. For this program, what will be the target clock rate for the computer B in GHz unit? [**1pts**]

ANS:

Execution time : $E_A = 10$ sec ; $E_B = 6$ sec

Number of cycle : $N_A = N$ sec ; $N_B = 1.2*N$ sec

Time period : $T_A = 1/4*10^9$ sec ; $T_B = T$

Therefore : $E_A / E_B = (N_A * T_A) / (N_B * T_B)$

$=> 10 / 6 = (N * T_A) / (1.2 * N * T)$

$=> T = (T_A * 6) / (10 * 1.2)$

$=> T = T_A / 2$

Hence frequency $F_B = 2 * F_A => F_B = (2*4)$ GHz = 8GHz

Clock rate of computer B is 8GHz

4. Consider the following piece of code in a 5-stage pipeline processor (as discussed in class).
   (a) Fill out the data hazard and resolution table. Use <inst#>-<stage> to denote instruction-stage in pipe line. For example, 1-MEM means 'MEM stage for instruction 1'. Mark the resolution methods as FWD (data forward) and STALL (stall). Consider no reordering in this case. [**2pts**]

   (b) Fill out data hazard and resolution table after the stall is inserted. [**1.5pts**]

   (c) Write down minimally re-ordered code to avoid stall. Fill out data hazard and resolution after re-ordering. [**1.5pts**]

#	**Instruction**	**Pipeline Stages**									
1	lw r1, r20, 0x2056	IF	ID	EXE	MEM	WB					
2	lw r2, r21, 0xF5C4		IF	ID	EXE	MEM	WB				
3	add r3, r1, r2			IF	ID	EXE	MEM	WB			
4	sw r3, r2, 0x451F				IF	ID	EXE	MEM	WB		
5	lw r4, r1, 0x0014					IF	ID	EXE	MEM	WB	
6	addi r8, r4, 0x1A						IF	ID	EXE	MEM	WB

ANS:

a) Data hazards and resolution table is as following.

Data Hazard & Resolution (Original)			
STAGE	DEPENDENCY	RESOLUTION	FWD-FROM
3-EXE	1-WB	FWD	1-MEM
3-EXE	2-WB	STALL	
4-EXE	2-WB	FWD	2-MEM
4-MEM	3-WB	FWD	3-EXE
6-EXE	5-WB	STALL	

b) Data hazards and resolution table after stall is as following.

Data Hazard (After Stall)			
STAGE	DEPENDENCY	RESOLUTION	FWD-FROM
3-EXE	2-WB	FWD	2-MEM
4-MEM	3-WB	FWD	3-EXE
6-EXE	5-WB	FWD	5-MEM

You may want to use the following tables for determining new dependencies and forward path after stall.

#	Instruction	Pipeline Stages											
1	lw r1, r20, 0x2056	IF	ID	EXE	MEM	WB							
2	lw r2, r21, 0xF5C4		IF	ID	EXE	MEM	WB						
X				O	O	O	O	O					
3	add r3, r1, r2				IF	ID	EXE	MEM	WB				
4	sw r3, r2, 0x451F					IF	ID	EXE	MEM	WB			
5	lw r4, r1, 0x0014						IF	ID	EXE	MEM	WB		
X								O	O	O	O	O	
6	addi r8, r4, 0x1A								IF	ID	EXE	MEM	WB

c) New re-ordered code is as following:

#	**Instruction**	**Pipeline Stages**									
1	lw r1, r20, 0x2056	IF	ID	EXE	MEM	WB					
2	lw r2, r21, 0xF5C4		IF	ID	EXE	MEM	WB				
3	lw r4, r1, 0x0014			IF	ID	EXE	MEM	WB			
4	add r3, r1, r2				IF	ID	EXE	MEM	WB		
5	sw r3, r2, 0x451F					IF	ID	EXE	MEM	WB	
6	addi r8, r4, 0x1A						IF	ID	EXE	MEM	WB

Data hazards and resolution table after code re-ordering is as following.

Data Hazard (After Code Reorder)			
STAGE	DEPENDENCY	RESOLUTION	FWD-FROM
3-EXE	1-WB	FWD	1-MEM
4-EXE	2-WB	FWD	2-MEM
5-MEM	4-WB	FWD	4-EXE

5. A 32-bit computer system uses virtual memory with 4KB page size. It uses a 4-way associative cache, which can hold 1MB information at a time with block size of 1KB. For any memory access, both TLB and cache access time is 1ns each. TLB miss rate is 25% with 20us penalty. Cache miss rate is 10% with 50us penalty. Page fault rate is 0.01% with page service (bring back page from disk to memory) 1s.

   (a) Determine number of bits used as page index and number of bits used as page number. [**1pts**]

   (b) Determine number of bits for tag, cache index and block index to access information from cache. [**1pts**]

   (c) How much time would be spent for one million (1,000,000) memory access in second unit (up to 3 decimal place) ? [**2pts**]

   (d) Determine TAG, CACHE LINE INDEX and BLOCK INDEX for an virtual address 0x8A2985B1 with TLB entry for 0x8A298 is 0x5BC81. [**1pts**]

ANS:

a) Page Index Bit = $\log_2(4K) = \log_2(2^2 * 2^{10}) = \log_2(2^{12}) = 12$
Therefore page number bit is (32 – 12) = 20

b) Number of Block index bit = $\log_2(4K) = \log_2(2^{10}) = 10$

Since cache can hold total 1MB information with 1KB block size, total number of cache line is (1MB / 1KB) = $(2^{20} / 2^{10}) = 2^{10} = 1024$. Since 4-way associative cache, cache line index should fully map (1024 / 4) = 256 lines. Hence number of cache line index is $\log_2(256)$ = 8 bits.

Therefore number of tag bits are (32 - 8 - 10) = 14.

c) There are multiple component of this problem.

- Without TLB or cache miss total access time is (1,000,000 * 2) ns = 2,000,000 ns = 0.002 sec
  - 1ns for TLB access and 1ns for cache access.
- Cache miss penalty = (1,000,000 * 0.1 * 50) us = 5,000,000 us = 5 sec
- TLB miss but no page fault penalty = (1,000,000 * 0.25 * 20) us = 5,000,000 us = 5 sec
- Page service time = (1,000,000 * 0.25 * 0.0001 * 1) sec = 25 sec
- Total time = (25 + 5 + 5 + 0.002) sec = 30.002 sec

d) Virtual Address 0x8A2985B1 has page number 0x8A298 (20-bits) and page index 0x5B1 (12-bits). Since 0x8A298 is mapped to 0x5BC81, physical address is 0x5BC815B1. Bit pattern of this address can split up into following way.

0101 1011 1100 1000 0001 0101 1011 0001

0001 0110 1111 0010 | 0000 0101 | 01 1011 0001

Therefore

- Tag : 0x16F2
- Cache Index : 0x05
- Block Index : 0x1B1

6. A NUMA system has 100 computation nodes with each node having total 128GB byte accessible local memory. Each node is a 64-bit machine, i.e. word size is 64-bit (information exchanged between processor and memory is 64-bit long). Each local memory access takes 10ns and each remote memory access takes 50ns.

   (a) What is the maximum amount of memory in TB unit, this NUMA system can offer to a program? [**1pts**]

   (b) A database program needs to access total 1TB in-memory data to process a query. Assume that this database program is the only program running on this NUMA system. What is the total memory access time in 'minute' unit (approximate it to two decimal place). [**2pts**]

   (c) One engineer enhanced this database program to be distributed – means 'n' number of this database program can be executed independently on different NUMA nodes accessing part of same 1TB in-memory data to process a single query. Additionally all these 'n' copy of the same database program access local memory only. However, we need to keep this number of parallel running database program to minimum to minimize inter process communication.

      i. What is the minimum number of 'n', assuming this is the only program running in the NUMA system? [**1pts**]

      ii. How much time, in 'minute' unit (approximate it to two decimal place), this distributed program will take to access whole 1TB memory ? [**1pts**]

ANS:

a) Maximum memory offered by this NUMA system is (100 * 128 GB) = 12800 GB = 12800 / 1024 TB = 12.5 TB.

b) Since NUMA is running only one program it will allocate total 128GB as local on one node and rest (1024 – 128) GB = 896 GB will be on remote node. Since 64-bit computer (and information is exchanged between processor and memory by 64-bits or 8 bytes)

- Total local memory access is $(128 * 2^{30}) / 8 = 2^{37} / 2^{3} = 2^{34}$ = 17,179,869,184 and it will take (17,179,869,184 * 10) ns = 2.86 min .
- Total remote memory access is $(896 * 2^{30}) / 8$ = 120,259,084,288 and it will take (120,259,084,288 * 50) ns = 100.22 min.
- Total 1TB access time will be (100.22 + 2.86)min = 103.08 min

c) i) Since this is the only program running on NUMA system, each process of the distributed set can occupy whole 128GB per node. Additionally each process access only local data. Therefore we need minimum (total memory to access / max local memory) = (1TB/128GB) = 8 number of process independently running on 8 nodes. Hence n = 8.

ii) Since all the process running in parallel accessing local memory access to 1TB memory will be done by the time that a single process accessing 128GB local memory (all of them will complete in parallel to each other). This will take total (128 GB / 8B) * 10 ns = 2.86 min.

Extra Credit Question

7. Your Project II processor P1 is modified to hook into a cache memory system (i.e. states are introduced to handle cache miss and handle mismatching access time for cache access) and running on 500MHz clock. A computing system C1 is using this P1 connected to a memory system M1 with separate instruction and data cache. Both these caches run on 500MHz clock and complete read / write operation using 1 cycle with cache miss penalty of 150ns for read miss and 200ns for write miss. The processor is now upgraded to run with 1.5GHz clock and work with same memory system M1. Let's call this processor as P2. A new system C2 uses processor P2 and memory system M1. A benchmark program B causes 5% miss rate for instruction cache and 10% miss rate for data cache. To run B, a processor needs to execute total 9 million instructions (9,000,000) of which 45% is data memory access. Also, 40% of the data memory access is write operation. What is the effective performance improvement in C2 (in terms of ratio) w.r.t. B? [**5pts**]

ANS:

- For system C1, since processor clock and memory clock are running at same speed, there is no additional wait for 5 stage execution of instruction in P1 for a always cache hit scenario. Hence CPI will be 5 for P1.
  - To run B without any miss, C1 will take (5 * (1/500MHz) * 9,000,000) = 90 ms.
  - Total instruction cache miss penalty is (9,000,000 * 0.05 * 150ns) = 67.5 ms. (it will be all read miss)
  - Total data read miss penalty will be (9,000,000 * 0.45 * 0.6 * 0.1 * 150ns) = 36.45 ms
  - Total data write miss penalty will be (9,000,000 * 0.45 * 0.4 * 0.1 * 200ns) = 32.4 ms.
  - Hence total execution time will be (90+67.5+36.45+32.4)ms = 226.25 ms.
- For system C2, since processor clock is running (1.5GHz / 500 MHz) = 3 times faster than memory clock, Each one clock cycle of memory will take same amount of time that of processor P2's 3 clock cycle. Therefore each memory operation (read or write) will take 3 cycle w.r.t. Processor P2's clock. Hence IF stage must be extended to 3 clock cycle and MEM stage must be extended to 3 clock cycle for P2. This means CPI for P2 will be (3+1+1+3+1) = 9.
  - To run B without any miss, C2 will take (9 * (1/1.5GHz) * 9,000,000) = 54 ms
  - Rest of the penalty calculation will remain same.
  - Hence total execution time will be (54+67.5+36.45+32.4)ms = 190.35 ms.
- Hence performance improvement in C2 is (226.25/190.35) = 1.19