[CPE223] COMPUTER ARCHITECTURES

M2 Mock Exam (ARM Ultimate Edition)

Time: 3 Hours | Full Score: 60 Points

Part 1: Multiple Choice Questions (20 Points)

Instructions: Select the correct answer. (1 point each)

1. Interrupt Nesting can be handled using which of the following scheme?

A. Vectored Interrupt

B. Daisy Chain

C. Priority

D. All of the above

E. None of the above

2. Which statement is not true about expansion bus?

A. SCSI allows overlapping of data transfer request.

B. USB Hubs are an intermediate control point between a host and I/O devices.

C. In PCI device, there is a small configuration RAM that stores information about the device.

D. All are true.

E. All are false.

3. Which is not a function of an I/O interface circuit?

A. Provides a storage buffer for at least one word of data.

B. Performs any format conversion that may be necessary to transfer data between the bus and the I/O device.

C. Provides an address-decoding circuit to determine when being addressed by the processor.

D. Contains status flags that can be accessed by the processor.

E. Generates a clock signal to synchronize the I/O device and the processor.

4. What is not correct for I/O operation?

A. The processor checks the status flag repeatedly in polling.

B. After reading data, the SIN flag is set to 1 to indicate it's ready for new data.

C. An Interrupt Service Routine (ISR) is called when an interrupt occurs.

D. The processor responds to an interrupt with the INTA signal.

E. Keyboard and Display usually have separate status registers.

5. Which signal is NOT important in a polling I/O operation?

A. SIN

B. SOUT

C. DATAIN

D. DIRQ (Device Interrupt Request)

E. STATUS

6. In a Carry Look-Ahead Adder (CLA), which statement perfectly describes its practical implementation?

A. CLA is completely independent of the number of bits, so a 64-bit CLA has the same gate delay as a 4-bit CLA.

B. CLA calculates all carry signals simultaneously by completely eliminating the G_i and P_i dependencies.

C. A purely flat CLA for 32-bit operands is highly impractical due to fan-in limitations of logic gates, requiring a cascaded approach.

D. CLA is widely used in multiplication but cannot be used for subtraction.

E. CLA uses fewer logic gates and is vastly cheaper than Carry Ripple Adder (CRA).

7. When executing the Booth Algorithm, which multiplier bit-pattern below will cause the algorithm to perform WORSE than the standard shift-and-add algorithm?

A. 00001111

B. 11110000

C. 11111111

D. 01010101

E. 10000001

8. Consider the Non-Restoring Division algorithm. If the current remainder in the accumulator (A) is NEGATIVE after a subtraction, what is the exact sequence of actions for the NEXT cycle?

A. Shift left, then Subtract divisor, then set $q_0 = 0$.

B. Shift left, then Add divisor, then set q_0 based on the new sign of A.

C. Restore A by adding divisor, Shift left, then Subtract divisor.

D. Set $q_0 = 0$, Shift left, then Add divisor.

E. Shift right, then Add divisor, then set $q_0 = 1$.

9. How many gate delays are required to compute the final summation bit (s_{31}) of a 32-bit CLA adder built from cascaded 4-bit CLA adder blocks?

A. 10 gate delays

B. 14 gate delays

C. 17 gate delays

D. 18 gate delays

E. 32 gate delays

10. To increase the efficiency of the Booth algorithm, Bit-pair recoding is used. The recoded pair `(+1, -1)` is mathematically equivalent to which single operation?

A. (0, -1)

B. (0, +1)

C. (+1, 0)

D. (-1, 0)

E. (0, +2)

11. Which design decision makes the ARM ISA appropriate for pipelining implementation?

A. ARM instructions have highly variable lengths to save memory space.

B. ARM ISA contains a large number of complex memory-to-memory operations.

C. ARM instructions are uniformly 32-bit (in standard mode), facilitating a streamlined fetch and decode stage.

D. ARM relies heavily on direct hardware interrupt manipulation.

E. ARM processors do not use a Program Counter (PC).

12. Which of the following statements is TRUE according to the ARM (RISC) architecture?

A. Arithmetic operations can be performed directly on variables stored in main memory.

B. Load and Store instructions (LDR/STR) must be used to move data between memory and registers before performing arithmetic operations.

C. The processor has only one general-purpose register.

D. The PC register holds the operand currently being executed.

E. The IR register holds the memory address to be accessed.

13. According to the Single Bus CPU Architecture, what is the sequence of control signals for the FIRST step of any Instruction Fetch?

A. $Z_{out}, PC_{in}, WMFC$

B. MDR_{out}, IR_{in}

C. $PC_{out}, Y_{in}, WMFC$

D. $PC_{out}, MAR_{in}, Read, Select4, Add, Z_{in}$

E. $R1_{out}, MAR_{in}, Read$

14. In a Single-Bus CPU architecture, why is the `Y` register necessary during an ALU operation (e.g., `ADD R1, R2, R3`)?

A. To store the final result of the ALU before transferring to Z.

B. To hold the instruction opcode for the Control Unit.

C. To temporarily hold one of the operands for the ALU, since the internal bus can only carry one word at a time.

D. To interface directly with the data lines of the memory bus.

E. To store the effective address of the operand.

15. What does the control signal `WMFC` stand for?

A. Write Memory Function Complete

B. Wait for Memory Function Complete

C. Word Memory Fetch Cycle

D. Write Memory Fast Cycle

E. Window Management For CPU

16. Which technique can be used to reduce the effect of Control Hazard?

A. Operand Forwarding

B. Data Bypassing

C. Branch Prediction

D. Increasing memory size

E. Loop Unrolling

17. A situation where an instruction cannot proceed because it needs the result of a previous instruction that has not yet finished is called:

A. Structural Hazard

B. Control Hazard

C. Data Hazard

D. Cache Miss

E. Pipeline Stall

18. A Structural Hazard occurs in a pipeline when:

A. An instruction depends on a previous data calculation.

B. A branch instruction changes the PC dynamically.

C. Two instructions request to use the same hardware resource (e.g., Memory or Register File) at the same clock cycle.

D. The memory is too slow to respond to the processor.

E. An interrupt occurs during the execution stage.

19. In a 4-State Branch Prediction algorithm, if the current state is `LT` (Likely Taken) and the actual branch outcome is `Branch Not Taken` (BNT), what will be the next state?

A. ST (Strongly Taken)

B. LNT (Likely Not Taken)

C. SNT (Strongly Not Taken)

D. LT (Likely Taken)

E. Back to initial state

20. What is the primary purpose of a ”Branch Delay Slot”?

A. To stall the pipeline completely to ensure accuracy.

B. To place an instruction that must be executed anyway immediately after a branch, thus preventing a wasted clock cycle (flush).

C. To flush the wrong instructions from the pipeline automatically.

D. To delay the memory write-back.

E. To resolve data dependencies using software.

Part 2: Short Answer (10 Points)

Instructions: Briefly answer the following questions based on Lectures 5-8.

1. What is ”Interrupt Latency”?

---

2. What is the main disadvantage of Program-controlled I/O (Polling) regarding processor time?

---

3. What is the main difference between a Synchronous Bus and an Asynchronous Bus?

---

4. What does ”ISR” stand for?

---

5. What is ”Daisy Chain” used for in an interrupt system?

---

6. Why is Carry Look-Ahead Adder (CLA) faster than Carry Ripple Adder (CRA)?

---

7. What causes a ”Data Hazard” in pipelining?

---

8. Explain the concept of ”Operand Forwarding” (Data Bypassing).

---

9. What causes a ”Control Hazard”?

---

10. What is a ”Superscalar” processor?

Part 3: Problem Solving (30 Points)

Instructions: Show your work in detail.

Question 1: I/O Organization - Asynchronous Bus (7.5 Points)

From the timing diagram of an asynchronous bus protocol for an input transfer (Master reading from Slave) shown in class, answer the following:

1.1 (2.5 points) Why do we need the delay time between t_0 to t_1?

1.2 (2.5 points) What is the meaning of the action at t_4 (when Slave-ready drops from 1 to 0)?

1.3 (2.5 points) What happen, if we don't have the delay time between t_3 to t_4?

Question 2: Computer Arithmetic - Restoring Division (7.5 Points)

From the restoring-division circuit: **divide 13 by 7** using the circuit and show the content of register A and Q in each cycle.

(Note: Dividend Q = 13 (1101), Divisor M = 7 (00111), $-M$ in 2's complement = 11001)

A (5-bit)	Q (4-bit)	End of
00000	1101	Initial Values
00001	1010	First cycle *(Shift $\longrightarrow$ Sub $\longrightarrow$ Neg $\longrightarrow$ q0=0 $\longrightarrow$ Restore)*
00011	0100	Second cycle *(Shift $\longrightarrow$ Sub $\longrightarrow$ Neg $\longrightarrow$ q0=0 $\longrightarrow$ Restore)*
00110	1000	Third cycle *(Shift $\longrightarrow$ Sub $\longrightarrow$ Neg $\longrightarrow$ q0=0 $\longrightarrow$ Restore)*
00101	0001	Fourth cycle *(Shift $\longrightarrow$ Sub $\longrightarrow$ Pos $\longrightarrow$ q0=1 $\longrightarrow$ No Restore)*

Result: Remainder = 00101 (5), Quotient = 0001 (1)

Question 3: Processing Unit - Datapath (ARM Edition) (7.5 Points)

According to the single bus CPU architecture, please show your understanding of the stages and data paths as the steps and actions for the following **ARM assembly instructions** (destination operand on the left-hand side).

Note: Assume the hardware has a constant 4 connected to the MUX.

`LDR R1, [R2, #4]` *(Load into R1 the value from memory address calculated by R2 + 4)*

`ADD R1, R2, [R3]` *(Add the value in R2 with the value in memory at address R3, then store in R1)*

Question 4: Pipelining - Diagram & Delay Slot (7.5 Points)

Consider the following code:

```
I1: Load #0, C
I2: Compare A, B
I3: Branch>0 Action1
I4: DEC C
I5: Branch NEXT
Action1:
I6: INC C
NEXT:
I7: Load #0, D
```

a) Assume that a branch-not-taken is predicted, and that the branch target is known after the execution (E) stage, draw a 4-stage pipeline (F-D-E-W) diagram when **A = 2 and B = 1**.

b) Does delay slot exist? If so, show how to eliminate it.

Answer Key & Detailed Explanation

Part 1: Multiple Choice

1. **D)** การทำ Interrupt ซ้อนกันต้องอาศัยการจัด Priority ซึ่งทำได้ผ่านฮาร์ดแวร์แบบ Daisy Chain และใช้ Vectored ชี้ตำแหน่งเพื่อความรวดเร็ว
2. **C)** ผิด เพราะข้อมูล Configuration ของอุปกรณ์ PCI ถูกเก็บใน ROM ไม่ใช่ RAM
3. **E)** I/O Interface ไม่ได้มีหน้าที่สร้างสัญญาณ Clock ให้ Processor
4. **B)** ผิด หลังจากอ่านข้อมูลเสร็จ สเตตัส SIN ต้องถูกรีเซ็ตเป็น 0 เพื่อรอรับข้อมูลใหม่
5. **D)** DIRQ ใช้สำหรับการแจ้งเตือนแบบ Interrupt ไม่ได้ใช้ในระบบ Polling
6. **C)** การสร้าง CLA แบบรวดเดียว 32-bit เป็นไปไม่ได้ในทางปฏิบัติเพราะติดข้อจำกัด Fan-in ของ Logic gate จึงต้องต่อแบบ Cascade ย่อยๆ
7. **D)** รูปแบบบิต 010101... สลับกันไปมาจะทำให้ต้องคำนวณบวก/ลบในทุกๆ รอบ ซึ่งเป็น Worst-case ของ Booth
8. **B)** กฎของ Non-restoring คือถ้ารอบที่แล้ว A ติดลบ รอบถัดไปให้ Shift left แล้วสลับไป "บวก (Add)" ตัวหารแทน
9. **D)** ใช้สูตร $3 + 2(n - 1)$ หา Carry out บล็อกสุดท้าย $3 + 2(7) = 17$ delays และบวกอีก 1 delay สำหรับหา Sum จะได้ 18 gate delays
10. **B)** อิงจากตารางความสัมพันธ์ Bit-pair recoding
11. **C)** รูปแบบคำสั่งที่ขนาดคงที่ 32-bit ทำให้ง่ายต่อการนำไปทำ Pipelining
12. **B)** สถาปัตยกรรมแบบ Load/Store ของ ARM ต้องโหลดค่าเข้า Register ก่อนจึงจะคำนวณผ่าน ALU ได้
13. **D)** ลำดับสัญญาณที่ถูกต้องในการ Fetch คำสั่ง
14. **C)** Bus มีเส้นเดียว ส่งข้อมูลได้ทีละตัว จึงต้องเอาตัวแปรแรกไปพักไว้ที่ Y ก่อน แล้วให้ Bus ส่งตัวแปรที่สองเข้า ALU
15. **B)** Wait for Memory Function Complete
16. **C)** Branch Prediction ใช้คาดเดาและแก้ Control Hazard ส่วน Forwarding ใช้แก้ Data Hazard
17. **C)** Data Dependency ทำให้เกิด Data Hazard
18. **C)** Structural Hazard คือการแย่งใช้ทรัพยากร (Resource) ตัวเดียวกันในเวลาเดียวกัน
19. **B)** เมื่อเดาว่า Taken แต่ผลออกมาคือ Not Taken สถานะจะถูกลงโทษลดระดับ 1 ขั้นไปเป็น LNT ทันที
20. **B)** เอาคำสั่งที่ยังไงก็ต้องทำงาน (ไม่ว่าเงื่อนไขจะเป็นจริงหรือไม่) มาใส่หลัง Branch เพื่อไม่ให้เสีย Clock Cycle ไปเปล่าๆ

Part 2: Short Answer

1. **Interrupt Latency:** The time between when an interrupt request is received and when the Interrupt Service Routine (ISR) starts execution. (ระยะเวลาตั้งแต่ระบบได้รับคำขอ Interrupt จนถึงตอนที่เริ่มรันโค้ด ISR)

2. **Disadvantage of Polling:** The processor spends a lot of time in wait loops checking the status flag, causing it to idle instead of performing useful computation. (CPU เสียเวลาไปกับการวนลูปเช็คสถานะเปล่าๆ)

3. **Synchronous vs Asynchronous Bus:** A Synchronous Bus uses a common clock signal for timing, while an Asynchronous Bus uses a handshake protocol (Master-ready, Slave-ready) to control data transfers.

4. **ISR:** Interrupt Service Routine

5. **Daisy Chain:** ใช้จัดลำดับความสำคัญ (Priority) ของอุปกรณ์ I/O ที่ขอ Interrupt มาพร้อมกัน อุปกรณ์ที่ต่อสาย INTA ใกล้ Processor ที่สุดจะได้รับสิทธิ์ก่อน

6. **CLA vs CRA:** CLA มีวงจรคำนวณค่า Carry Out ของแต่ละบิตล่วงหน้าแบบขนาน (Parallel) ทำให้ไม่ต้องเสียเวลารอ Carry ส่งต่อ (Ripple) จากบิตก่อนหน้าแบบ CRA

7. **Data Hazard:** เกิดจากการที่คำสั่งปัจจุบัน ต้องรอใช้ผลลัพธ์จากคำสั่งก่อนหน้าที่ยังประมวลผลหรือเขียนลง Register ไม่เสร็จ (Data Dependency)

8. **Operand Forwarding:** การดึงผลลัพธ์จากปลายทางของ ALU ส่งย้อนกลับไปเป็น Input ให้กับ ALU สำหรับคำสั่งถัดไปทันที โดยไม่ต้องรอให้ถึงขั้นตอน Write-back เพื่อช่วยแก้ปัญหา Data Hazard

9. **Control Hazard:** เกิดจากคำสั่งประเภท Branch/Jump ที่ทำให้ CPU ต้องเปลี่ยนเส้นทางการทำงาน ทำให้คำสั่งที่ถูกดึง (Fetch) มาแล้วใน Pipeline กลายเป็นคำสั่งที่ผิดและต้องถูก Flush ทิ้ง

10. **Superscalar:** โปรเซสเซอร์ที่มีหน่วยประมวลผลย่อย (Processing Units) หลายตัวทำงานขนานกัน ทำให้สามารถ Execute คำสั่งที่ต่างกันได้มากกว่า 1 คำสั่งใน Clock Cycle เดียวกัน

Part 3: Problem Solving

Question 1: I/O Organization - Asynchronous Bus

- **1.1 (t_0 to t_1 delay):** To allow time for the address and command signals to travel on the bus, stabilize, and give the device interface circuitry time to decode the address to know it is being accessed before the Master asserts the Master-ready signal.

- **1.2 (Action at t_4):** At t_4, the Slave detects that the Master-ready signal has dropped to 0 (meaning the Master has successfully received the data). The Slave responds by dropping Slave-ready to 0 and removing its data from the bus, signaling the completion of its part in the transfer.

- **1.3 (If no delay between t_3 to t_4):** If a new address and command are placed on the bus immediately before the old Master-ready signal properly decays to 0, devices might misinterpret the leftover signal as a new valid request, causing a data collision or reading error on the bus.

Question 2: Restoring Division

A (5-bit)	Q (4-bit)	End of
00000	1101	Initial Values
00001	1010	First cycle *(Shift ⟶ Sub ⟶ Neg ⟶ q0=0 ⟶ Restore)*
00011	0100	Second cycle *(Shift ⟶ Sub ⟶ Neg ⟶ q0=0 ⟶ Restore)*
00110	1000	Third cycle *(Shift ⟶ Sub ⟶ Neg ⟶ q0=0 ⟶ Restore)*
00101	0001	Fourth cycle *(Shift ⟶ Sub ⟶ Pos ⟶ q0=1 ⟶ No Restore)*

Result: Remainder = 00101 (5), Quotient = 0001 (1)

Question 3: Processing Unit - Datapath

- **`LDR R1, [R2, #4]`**

  1. $PC_{out}, MAR_{in}, Read, Select4, Add, Z_{in}$
  2. $Z_{out}, PC_{in}, WMFC$
  3. MDR_{out}, IR_{in}
  4. $R2_{out}, Select4, Add, Z_{in}$ *(Calculate Effective Address: R2 + 4)*
  5. $Z_{out}, MAR_{in}, Read$ *(Send the calculated address to MAR and Read)*
  6. $WMFC$ *(Wait for data from memory)*
  7. $MDR_{out}, R1_{in}, End$ *(Store the fetched data into R1)*

- **`ADD R1, R2, [R3]`**

  1. $PC_{out}, MAR_{in}, Read, Select4, Add, Z_{in}$
  2. $Z_{out}, PC_{in}, WMFC$
  3. MDR_{out}, IR_{in}
  4. $R3_{out}, MAR_{in}, Read$ *(Send address R3 to memory)*
  5. $WMFC$
  6. MDR_{out}, Y_{in} *(Hold the value from memory in register Y)*
  7. $R2_{out}, SelectY, Add, Z_{in}$ *(Add R2 to the value in Y)*
  8. $Z_{out}, R1_{in}, End$ *(Store the final result in R1)*

Question 4: Pipelining - Diagram & Delay Slot

a) Pipeline Diagram (A=2, B=1 ⟶ Branch Taken, but predicted Not Taken):

Instruction	1	2	3	4	5	6	7	8
I1	F	D	E	W				
I2		F	D	E	W			
I3			F	D	E	W		
I4				F	D	X		*(Flush)*
I6					F	D	E	W

b) Delay slot existence & Elimination:

Yes, a delay slot exists (the wasted cycle in clock 4 when flushing the incorrectly fetched I4).

How to eliminate: เราสามารถกำจัด Delay slot ได้โดยใช้เทคนิค **Instruction Reordering** (การสลับคำสั่ง) โดยย้ายคำสั่ง `I1: Load #0, C` ลงมาวางต่อท้าย `I3: Branch>0 Action1` (ให้เข้ามาอยู่ในพื้นที่ Delay Slot พอดี) เนื่องจากคำสั่ง `I1` ต้องถูก Execute เสมอไม่ว่าจะเข้าเงื่อนไข Branch หรือไม่ และไม่มีผลต่อตัวแปร A และ B ที่ใช้เปรียบเทียบใน I2 และ I3 การสลับนำ `I1` มาวางในจังหวะนี้จึงช่วยอุดรอยรั่วของ Clock Cycle ได้โดยไม่ทำให้โปรแกรมทำงานผิดพลาด