06-07 年

(1) True or False

F 1. (F)Structure and architecture can be considered as the same concept about the computer.
T 2. (T)Personal computer used in life is a computer system in fact
T 3. (T)Registers provides storage internal to CPU.
T 4. (T)Instruction set is an example of architectural attributes.
T 5. (T)A multiply instruction can be implemented by a special multiply unit or by repeated use of add unit, which is an issue of computer organization.
T 6. (T)CPU, main memory and I/O devices communicate each other by system bus.
T 7. (T)PC (program counter) contains the instruction address to be fetched and executed.
T ~~F~~ 8. (T)Computer always fetches instruction from memory cell directed by PC (program counter).
T 9. (T)Loops (such as 'for' loop in c language) are a kind of locality of program.
F 10. (F)Interrupts can interrupt the current executing instruction.
T 11. (T)Cache line and main memory block has the same size.
F 12. (F)For associative mapping, a main memory block can only map into the special cache line.
T 13. (T)At first hit ratio will increase with the block size increases, but it will decreases if the block size is too large.
F 14. (F)All of the addressing techniques need some operations of memory referencing.
T 15. (T)The base-register addressing also takes advantage of the locality of memory references.
F 16. (F)For a 6-stage pipeline, a six times of speed up in the execution of instructions is certainly possible. p.401 $\frac{6\cdot n}{6+(n-1)}$ $\frac{1}{\frac{1}{n}+1-\frac{1}{n}}$ =
T 17. (T)Output dependencies and antidependencies arise because the values in registers may no longer reflect the sequence of values dictated by the program flow.

true data: write-read
output : write-write
antidependency: read-write

F 18. (F)The horizontal microinstructions have less length than vertical microinstructions.
F ~~T~~ 19. (F)The window scheme in the use of a large register file provides an efficient organization for storing local scalar and global variables in registers. p.437
T 20. (T)By register renaming, the output dependency and antidependency can be eliminated.

(2) Fill in blanks

LRU FIFO LFU Random

1. There are four kinds of replacement algorithms, and they are (LRU,FIFO,LFU and Random).
2. Immediate addressing and register addressing have no main memory access without considering stack addressing.
3. For increasing instruction-level parallelism, five fundamental limitations to the parallelism must be coped: true data dependency W-R, procedural dependency, resource conflict, output dependency W-W, antidependency. R-W
4. Three hardware techniques can be used in the superscalar processor to enhance performance: duplication of resources, out-of-order issue, register renaming.

~~Machine parallelism instruction issue policy~~

duplication of resources
out-of-order issue
renaming

3. Compressive problem

1. Consider the following assembly- language program:

6

E: Execute. Calculates memory address.

C: Read from or write to the cache.

Note that data and instruction cache are separated and can be accessed simultaneously. There is only ONE functional unit for each stage. If an instruction needs an operand that is altered bye the preceding instruction, bypass technique is NOT used here for simplicity.

(a) If the program is to execute in a RISC computer WITHOUT delayed branch and delayed load, and any branch prediction. Exit state is accomplished by inserting NOOP instructions into instruction stream by the compiler when data dependencies or branch instructions are met. Rewrite the compiled program and draw the pipeline.

(b) If delayed branch and delayed load are used by the compiler, rewrite the compiled program and draw the pipeline.

0011000
1100111

16

```
(a) 200 LOAD   B←Memory
    201 NOOP
    202 STORE  Memory←B+1
    203 LOAD   C←Memory
    204 NOOP
    205 STORE  Memory←C+1
    206 SUB
    207 ~~JU~~ JUMPZ 302
    --
    302

(b) 200 LOAD   B←Memory
    201 LOAD   C←Memory
    202 STORE  Memory←B+1
    203 STORE  Memory←C+1
    204 JUMPZ  300
    205 SUB    X←B-1
    ⋮
    300
```

2 | 51 ... 1
2 | 25 ... 1
2 | 12 ... 0
2 | 6 ... 0
2 | 3 ... 1
1 ... 1

I1 : Move R3, R7 /R3<- R7/
I2 : Load R8, (R3) /R3<- Memory(R3)/
I3 : Add R3, R3,4 /R3<- (R3)+4/
I4 : Load R9, (R3) /R9<- Memory(R3)/
I5 : BLE R8, R9,L3 /Branch if (R9)>(R8)/

a. What dependencies exist in the program?

	Which dependency?
(I1,I2)	W-R True data
(I1,I3)	W-W output
(I2,I3)	R-W anti-
(I3,I4)	W-R
(I2,I5)	W-R
(I4,I5)	W-R

a. using register renaming technique to rewrite the above assembly- language program.
Ri (i=1 2 3 ...) represents logical (symbol) register
Ri a (or Ri b ....) with subscripts a b c and so on represents physical register in the CPU.

```
I1  MOVE  R3a, R7a
I2  LOAD  R8a, (R3a)
I3  ADD   R3b, R3a, 4
I4  LOAD  R9a, (R3b)
I5  BLE   R8a, R9a.
```

W-W between I1 I3 & R-W between I2, I3 have been eliminated.

2. For a control unit, it has two basic functions, sequencing and execution, the following is a micro-programmed control unit, please try to answer the following questions:

a. How to perform the sequencing function for the control unit? (please give a simple description)
b. How to execute the micro-instructions? (please give a simple description)
c. ISZ X is an instruction: increment and skip if zero, try to write the possible sequence of micro-operations, and try to write the complete microprograme executed by the control unit to fetch and execute the ISZ X instruction (direct addressing).

由 Flag, ALU signal 等确定.
Branch address field.
Branch } ZR
CAR ← (CAR)+1.

~~1. read signal ~~Fetch~~ Control Memory~~
~~2.~~

c. Fetch
t_1 MAR ← (IR(address))
t_2 MBR ← Memory
t_3 MBR ← (MBR)+1
t_4 Memory ← (MBR)
test If MBR=0 then PC ← routine addr

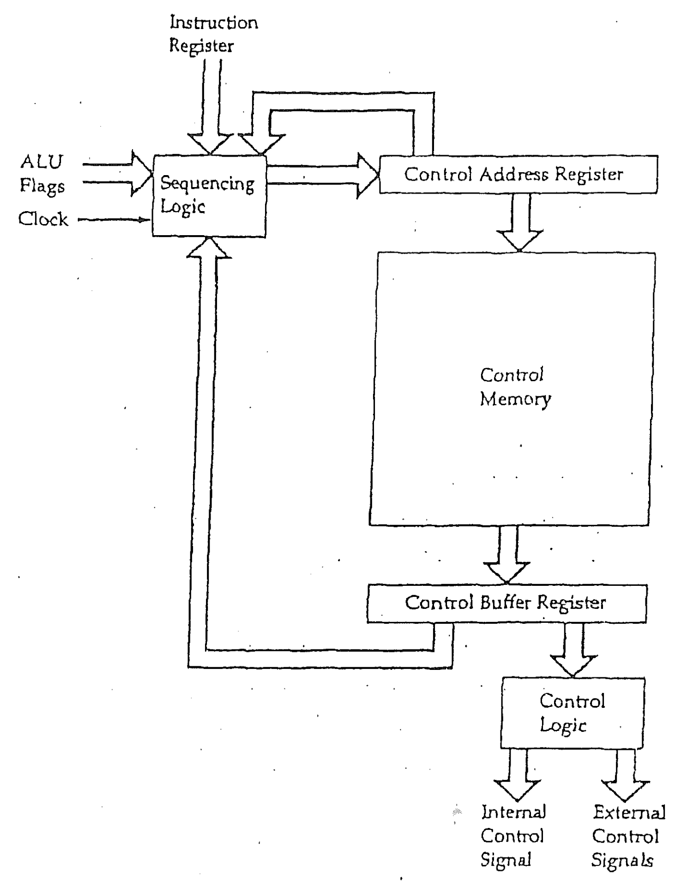

FIGURE 15.10. Control unit organization