PIPELINED PROCESSOR (I)

Assignment Project Exam Help

https://tutorcs.com

Lecturer: Hui Annie Guo

WeChat: cstutorcs

h.guo@unsw.edu.au

K17-501F

Lecture overview

- **Topics**
  - **Pipelining datapath**
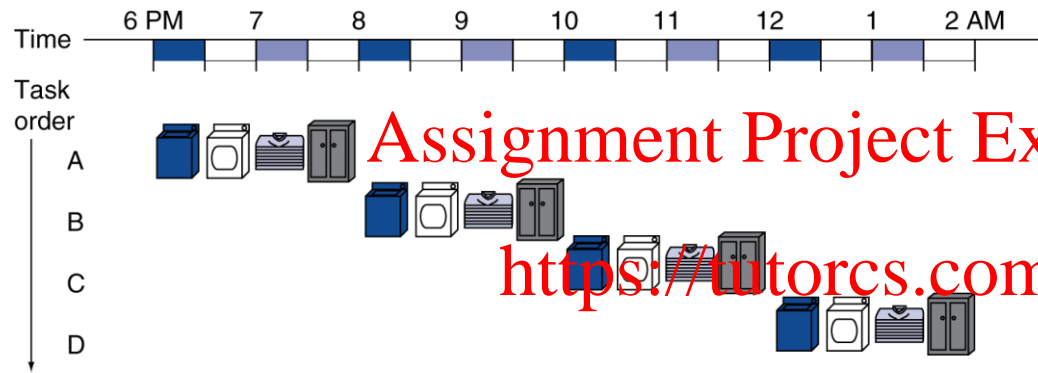 - **Pipelining control**

  Assignment Project Exam Help

  https://tutorcs.com

- **Suggested reading**

  WeChat: cstutorcs
  - **H&P Chapter 4.5 – 4.6**

Pipelining: example

- **Pipelined laundry**

Assignment Project Exam Help

https://tutorcs.com

WeChat: cstutorcs

- **Four loads:**
  - **Speedup = 8/3.5 = 2.3**
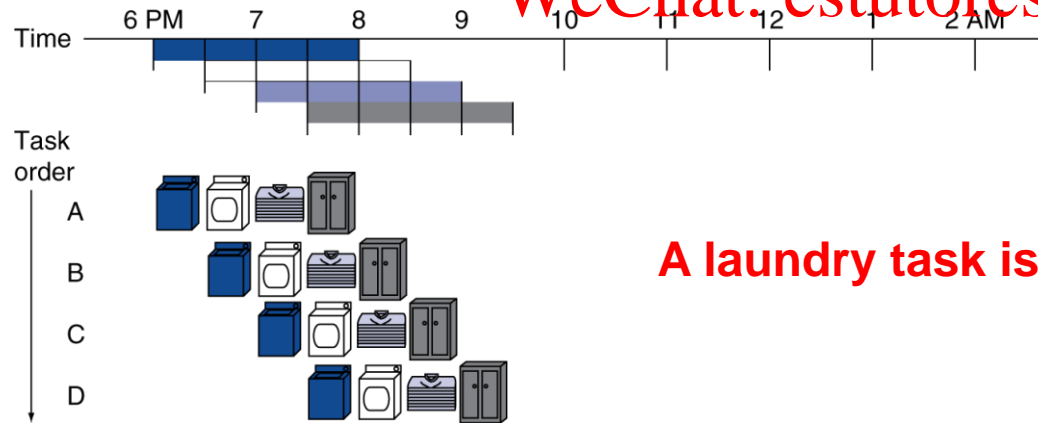
A laundry task is pipelined into four stages

MIPS pipelined datapath

Five stages:

1. **IF: Instruction fetch from memory**
2. **ID: Instruction decode & register read**
3. **EX: Execute operation or calculate address**
4. **MEM: Access memory operand**
5. **WB: Write result back to register**

MIPS ISA is designed for pipelining

- **All instructions are 32-bits**
  - **Easier to fetch in one cycle**
  - **cf. x86: 1- to 17-byte instructions**
- **Few and regular instruction formats**
  - **Can decode and read registers in one step**
- **Load/store addressing**
  - **Can calculate address in an early stage, access memory in a later stage**
    - **Since ALU available**
- **Alignment of memory operands**
  - **Memory access takes "one cycle"**
- **Most resources are available**

Pipelined instruction execution

time(cc)

Inst. #	1	2	3	4	5	6	7	8
Inst. i	IF	ID	EX	MEM	WB			
Inst. i+1		IF	ID	EX	MEM	WB		
Inst. i+2			IF	ID	EX	MEM	WB	
Inst. i+3				IF	ID	EX	MEM	WB

- **In each cycle, new instruction fetched and begins 5 cycle execution**

- **Ideally, the pipeline completes one instruction every clock cycle**

  - **Namely, 5 times faster if** *Tpipe = (1/5)*Tsc*

pipelined processor

single-cycle processor

Assignment Project Exam Help

https://tutorcs.com

WeChat: cstutorcs

Create pipeline stages (1)

- **Partition the single cycle datapath into sections. Each section forms a pipeline stage**

- **Considerations**

  - **Try to balance the stages for similar stage delays**

    - **To reduce the clock cycle time**

  - **Try to make data flow in the same direction**

    - **To avoid pipeline hazards**

      - Will be covered later

  - **Make sure the correct data to be used for each instruction in the pipeline**

    - **For correct functions**

Assignment Project Exam Help

https://tutorcs.com
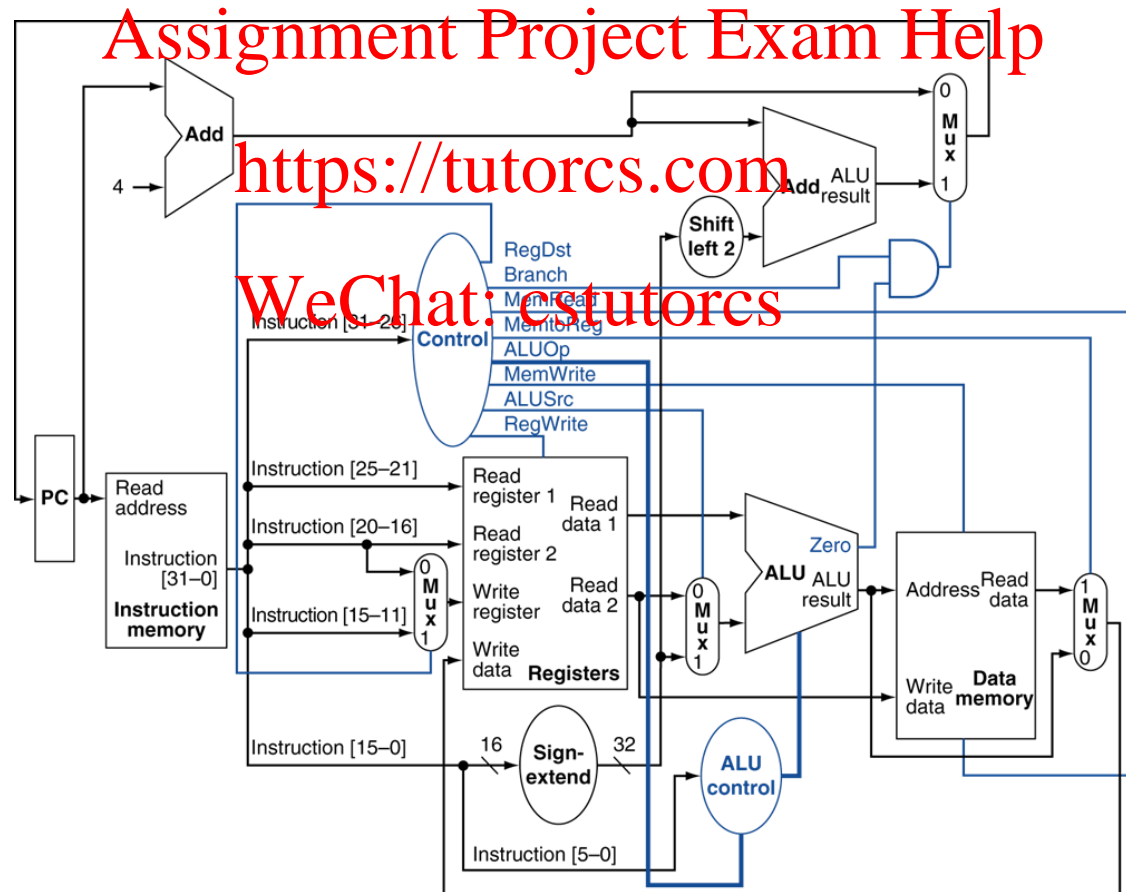
WeChat: cstutorcs

Partition the SC datapath

- **Try to balance the stages for a small clock cycle time**
  - **The function of each section is already specified**
    IF, ID, Ex, Mem, WB
  - **But we can tune the partition for a better solution**

Partition the SC datapath

- **Try to make data flow in the same direction**

IF → ID → Ex → Mem → WB ?
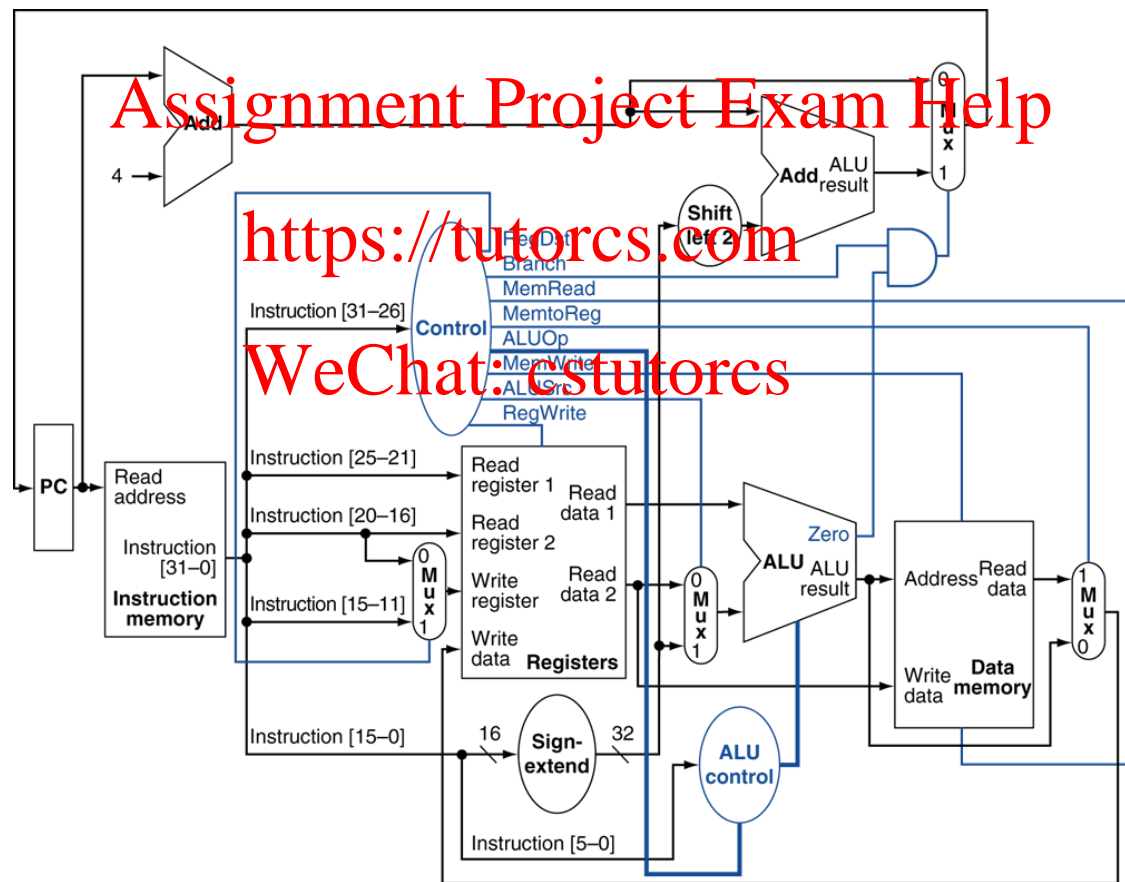
Partitioned five stages
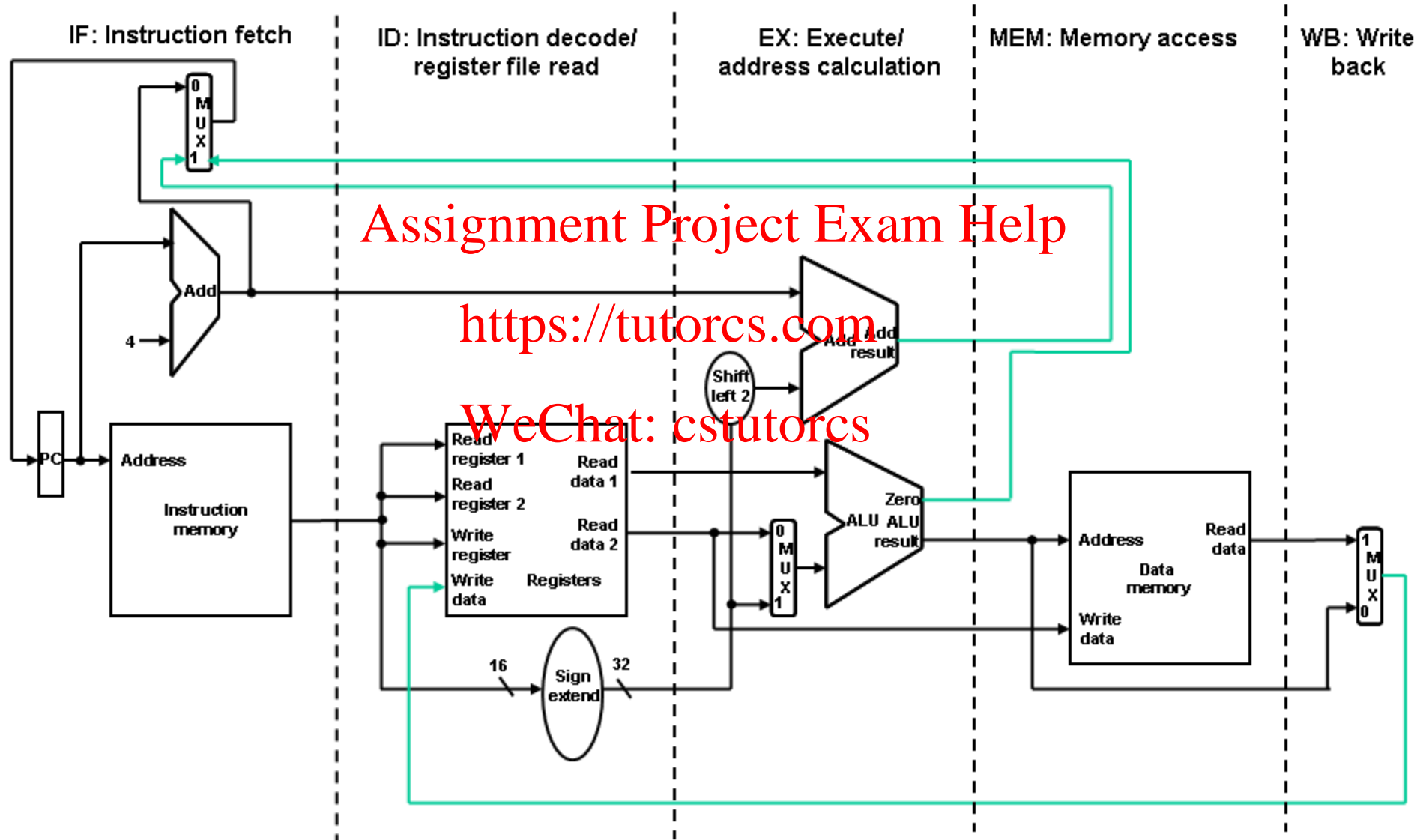
Stage representation
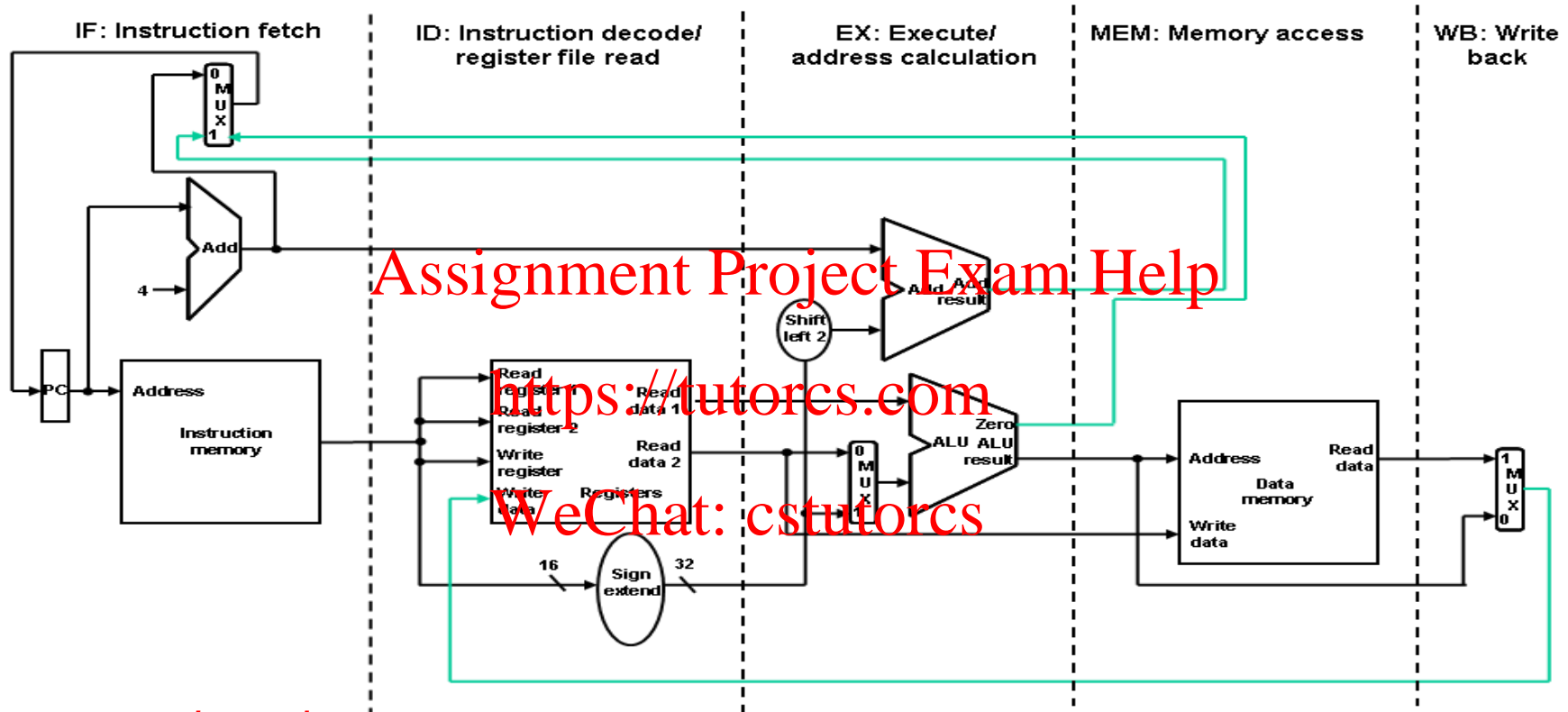
component based

operation based

IF   ID   Ex   Mem   WB
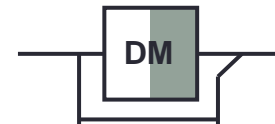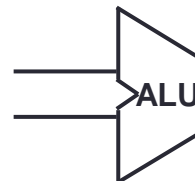
Execution timing diagram

- **Example 1: using major component to represent each stage**
  - **The *right half* of a register (memory) is highlighted as it is *read* and the *left half* is highlighted as it is written**

Assignment Project Exam Help

https://tutorcs.com

WeChat: cstutorcs
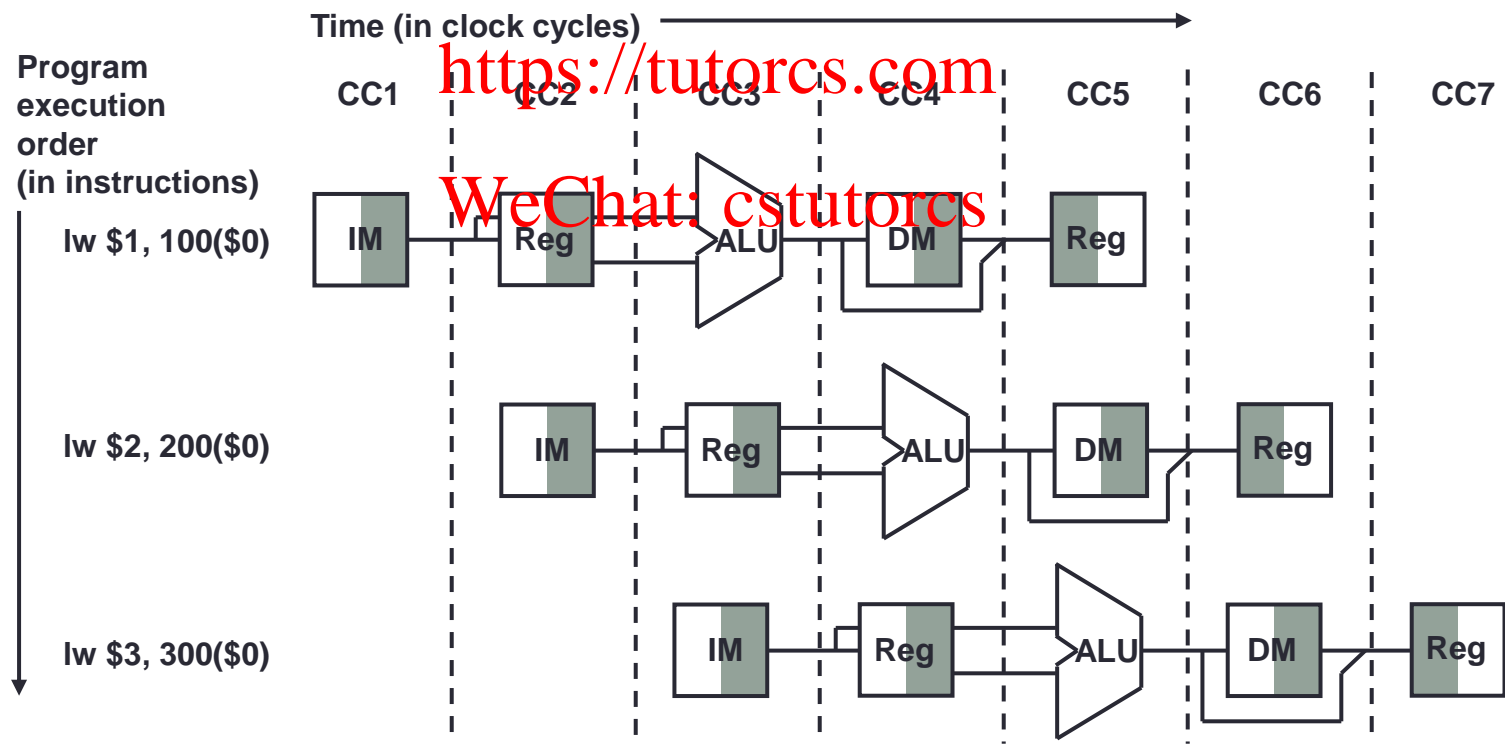
Execution timing diagram

- **Example 2: using operation to represent each stage**

Time

Assignment Project Exam Help

https://tutorcs.com

WeChat: cstutorcs

Program Flow
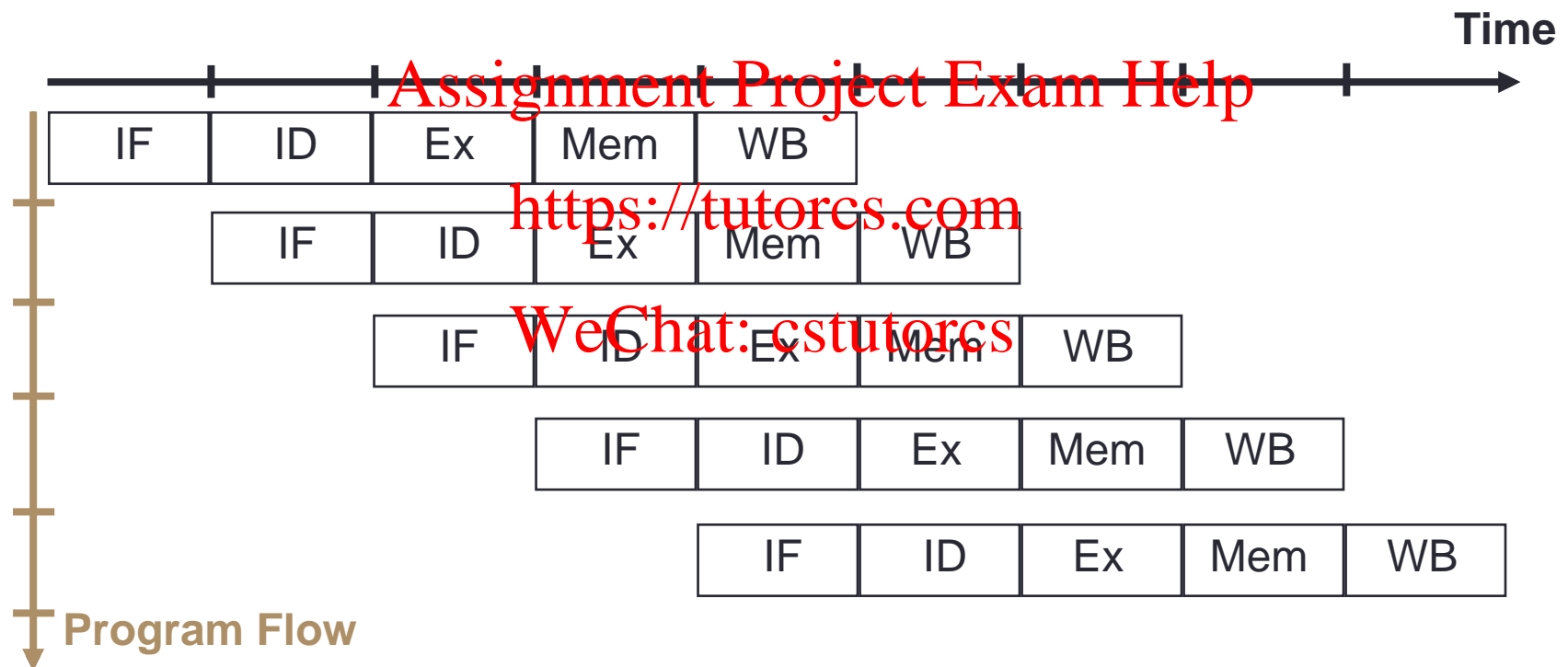
Create pipeline stages (2)

- **To retain the values of an instruction to be used in other stages, the values must be saved in registers.**

Assignment Project Exam Help

https://tutorcs.com

WeChat: cstutorcs
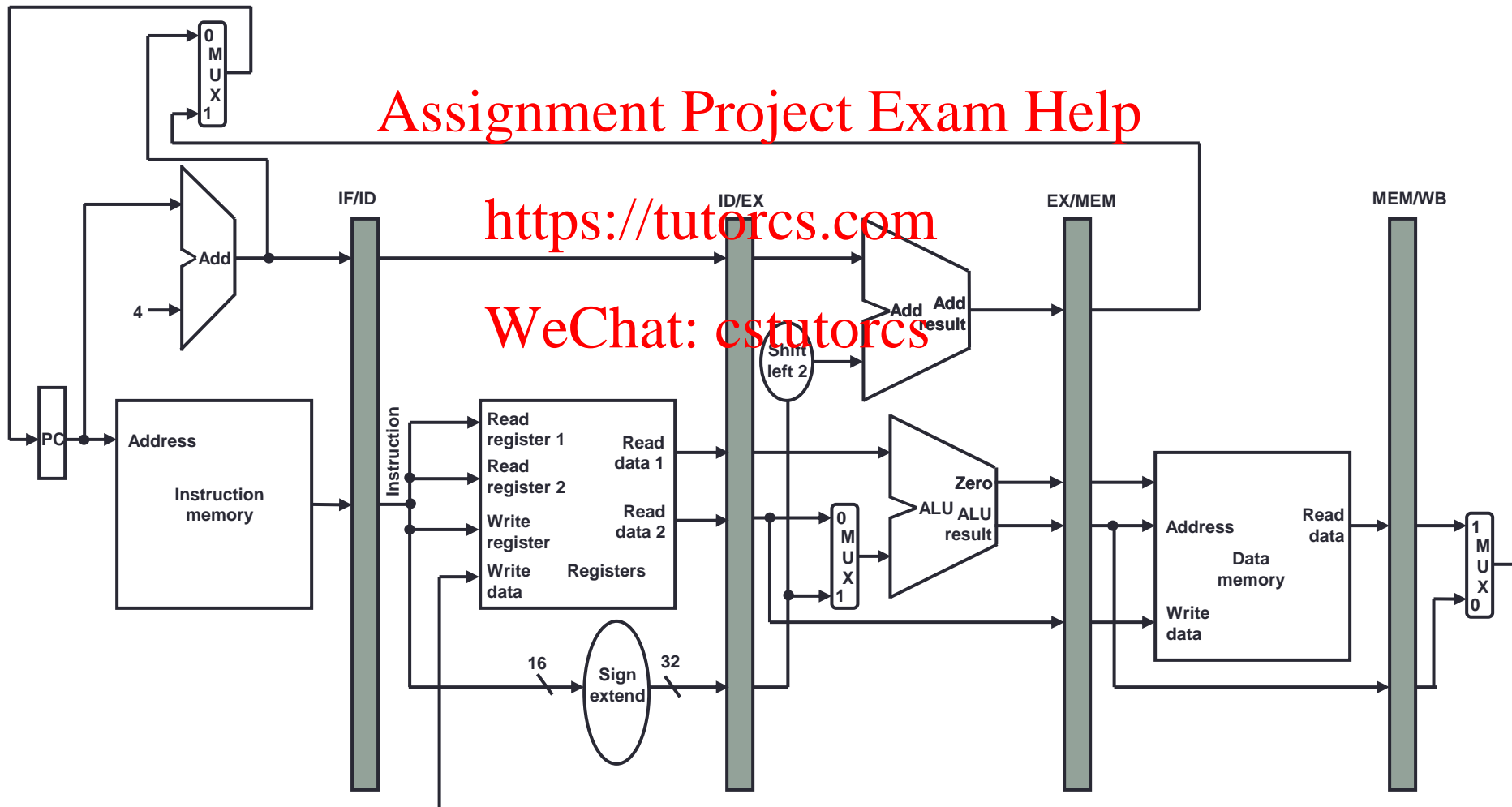
Notes on the pipelined datapath

- **The registers inserted between the pipeline stages are called pipeline stage register, or simply pipeline register.**

- **Each pipeline register is named after the two stages separated by that register**

  - **E.g. The register separating IF and ID stages is called the IF/ID register**

- **All instructions advance from one stage to the next stage**

  - **in one clock cycle**

Notes on the pipelined datapath (cont.)

- **The write back stage does not need a separate register since the register file is updated at the end of that stage**
  - **A separate register would be redundant.**
- **An example of pipeline execution is given in the following slides.**

Pipelined **LW** execution: **IF stage**
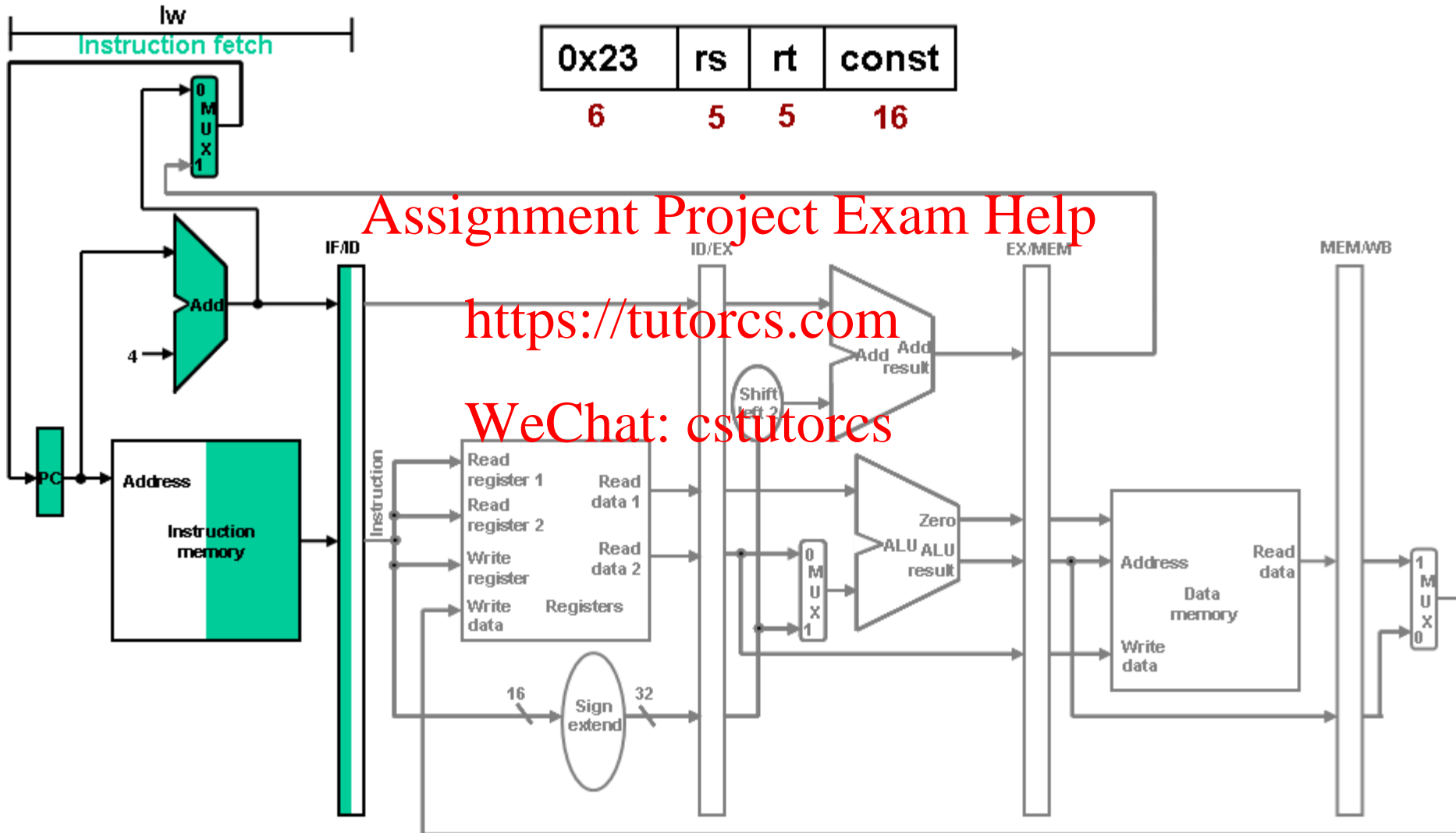
Pipelined LW execution: ID stage

0x23	rs	rt	const
6	5	5	16
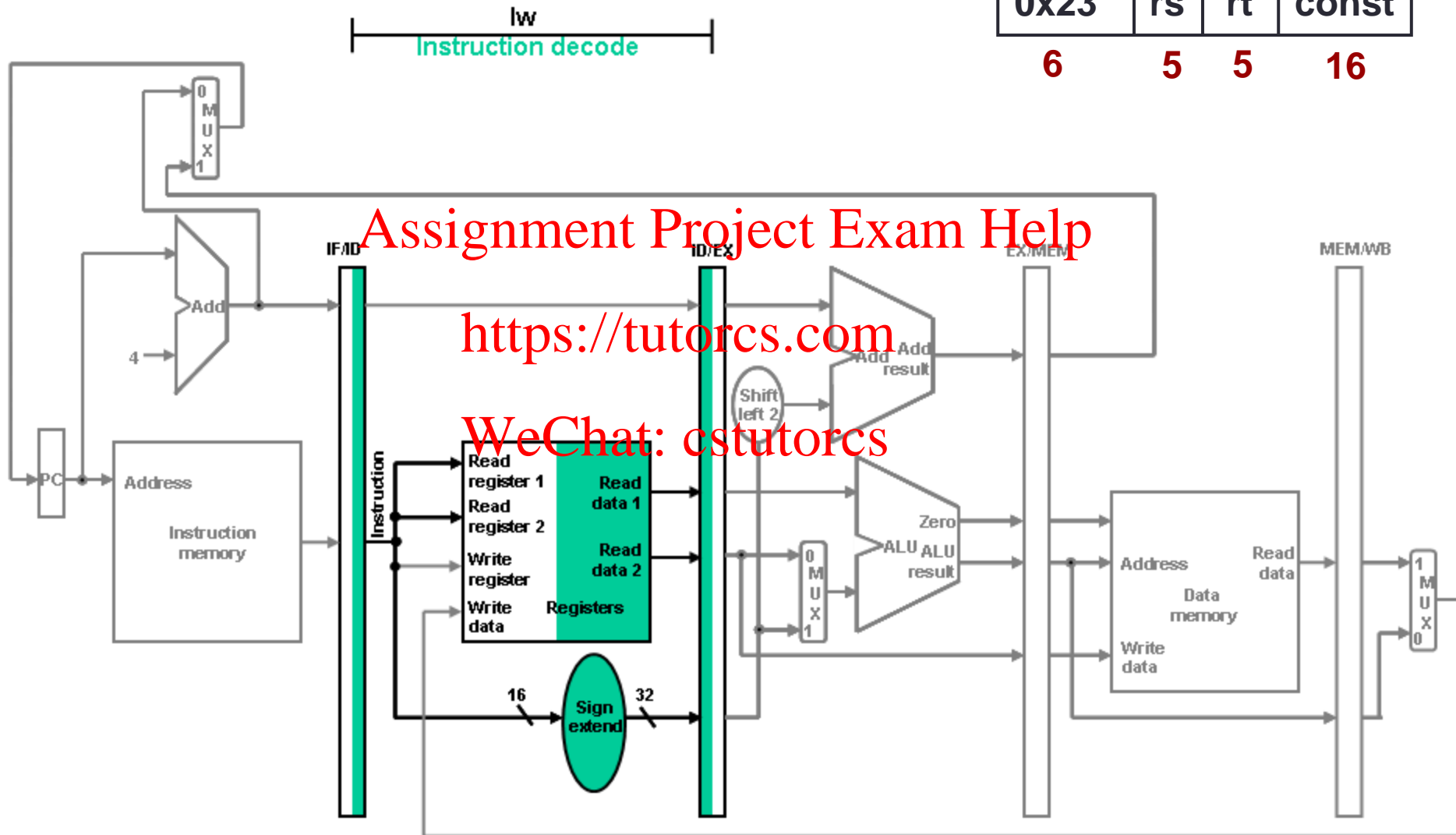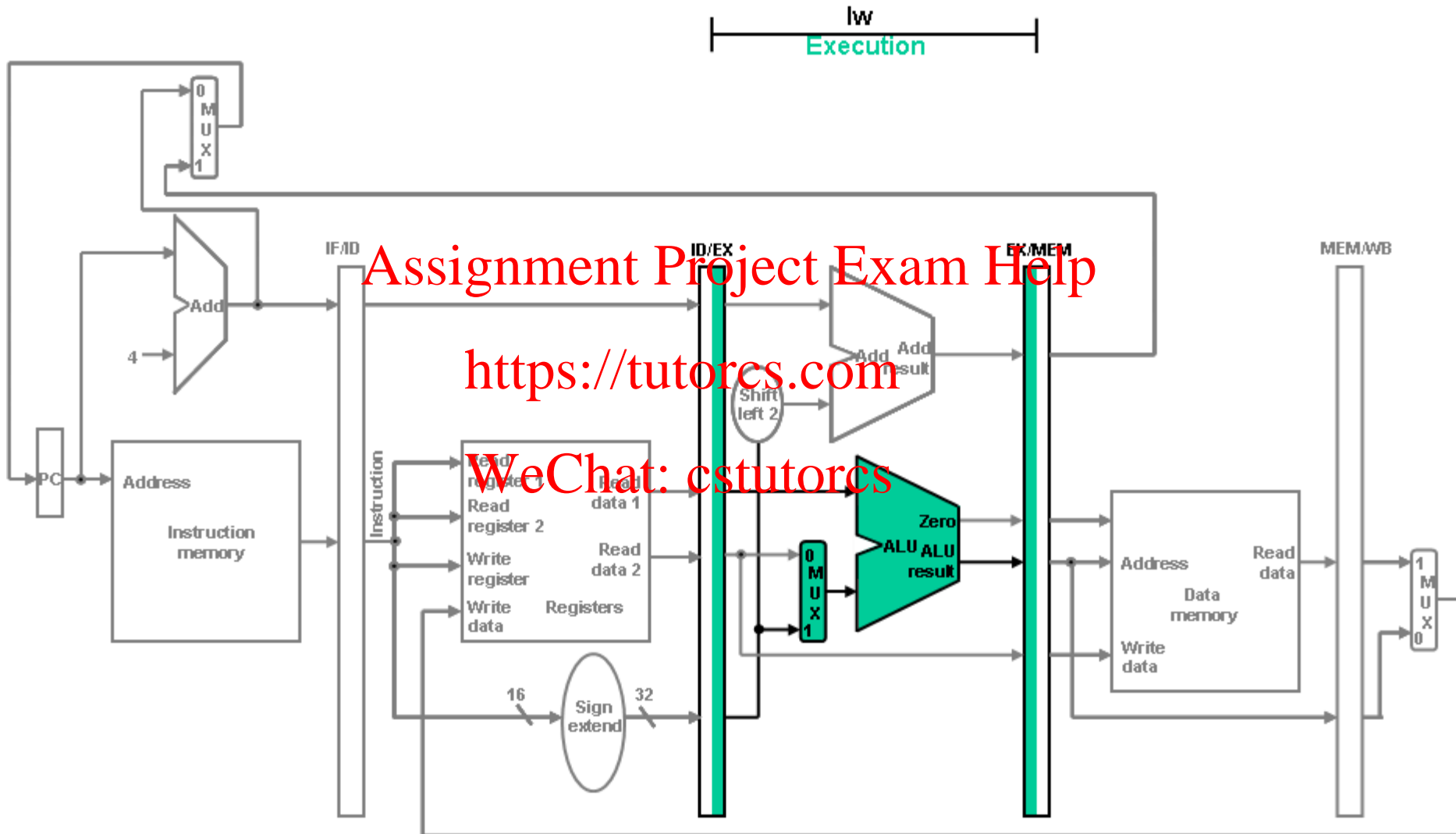
lw
Instruction decode

Assignment Project Exam Help

https://tutorcs.com

WeChat: cstutorcs

Pipelined LW execution: Ex stage

Assignment Project Exam Help

https://tutorcs.com

WeChat: cstutorcs

Pipelined LW execution: Mem stage

Assignment Project Exam Help

https://tutorcs.com

WeChat: cstutorcs

Pipelined LW Execution: WB stage

Assignment Project Exam Help

https://tutorcs.com

WeChat: cstutorcs
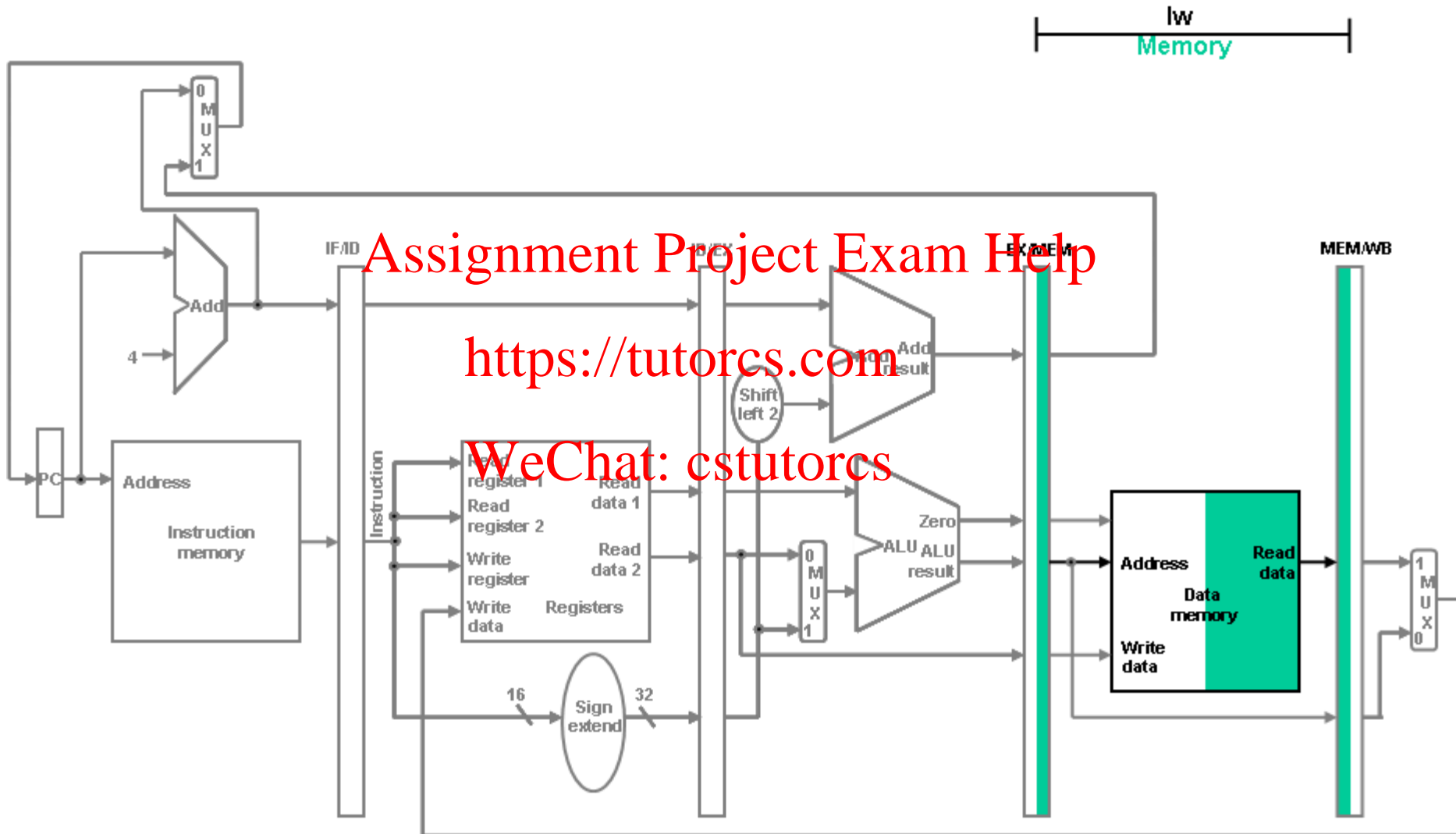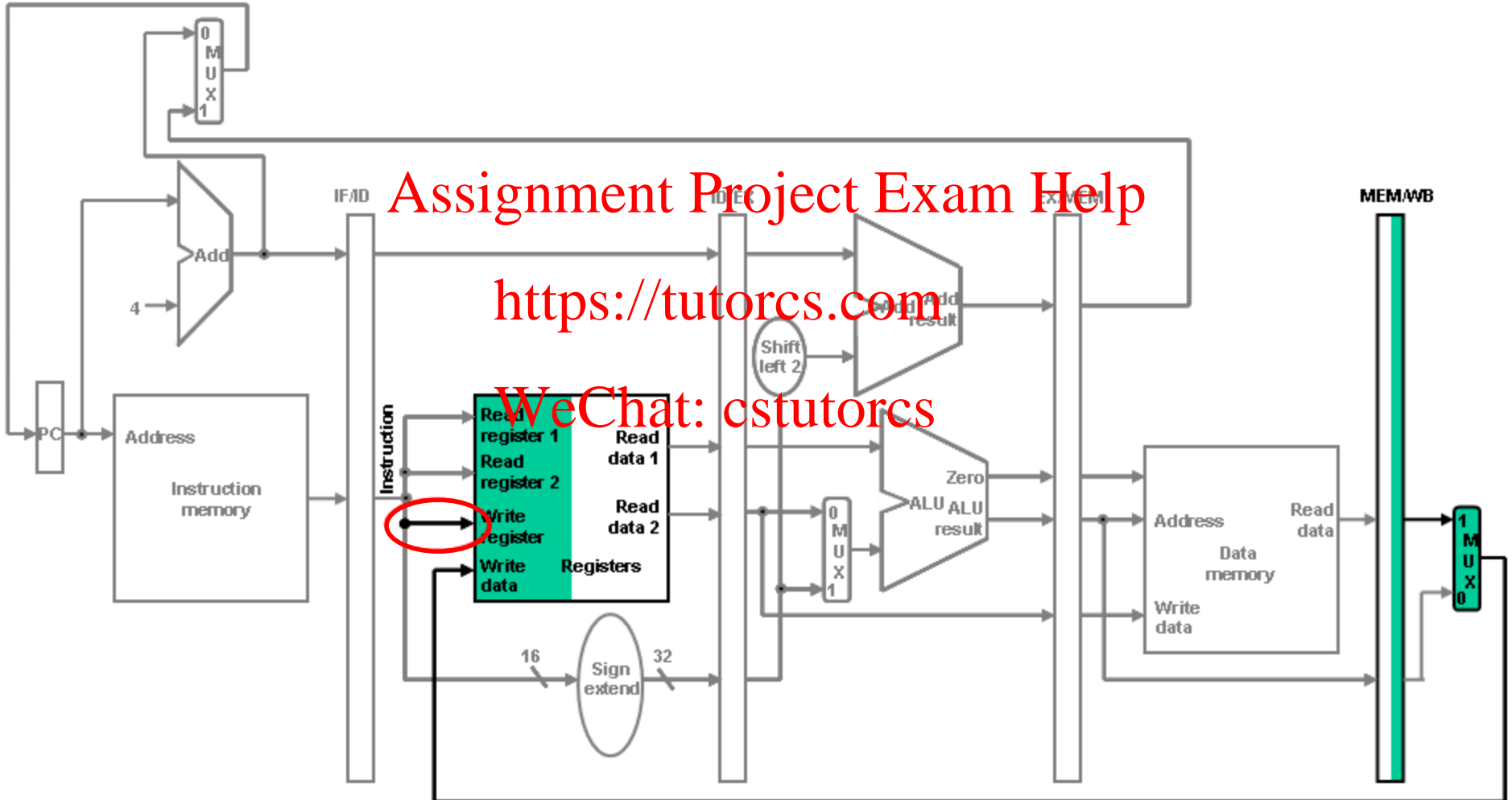
Notes on pipelined **LW** execution

- **The previous five slides show the active portions of the datapath as a load instruction progresses through the pipeline**

- **The operations of the five stages are as follows:**

  - *Instruction fetch*
    - **The instruction is read from memory pointed by PC and is stored in IF/ID. PC is incremented by 4 and written back ready for the next instruction. The new PC value is stored in IF/ID for instructions such as _BEQ_.**
    - We store all data that may be needed by subsequent stages.

Assignment Project Exam Help

https://tutorcs.com

WeChat: cstutorcs

Notes on pipelined LW execution (cont.)

- *Instruction decode and register file read*

  - **The immediate field is retrieved from IF/ID and sign extended;**

  - **The two source registers are read;**

  - **All three values are stored in ID/EX along with everything else that may be needed by the instruction in the next clock cycle.**

- *Execution or address calculation*

  - **The value in register 1 and the sign-extended immediate value from ID/EX are added by ALU and the result is placed in the EX/MEM register.**

- *Memory access*

  - **Data memory is read using the address from EX/MEM; The read data is loaded into the MEM/WB register.**

- *Write back*

  - **The data from MEM/WB is written to the register file.**

Assignment Project Exam Help

https://tutorcs.com

WeChat: cstutorcs

Pipelined SW execution: IF stage

Assignment Project Exam Help

https://tutorcs.com

WeChat: cstutorcs

Pipelined SW execution: ID stage

Pipelined SW execution: Ex stage

sw rt, const(rs)   # Mem(rs+const) ← rt

Pipelined SW execution: Mem stage

Assignment Project Exam Help

https://tutorcs.com

WeChat: cstutorcs
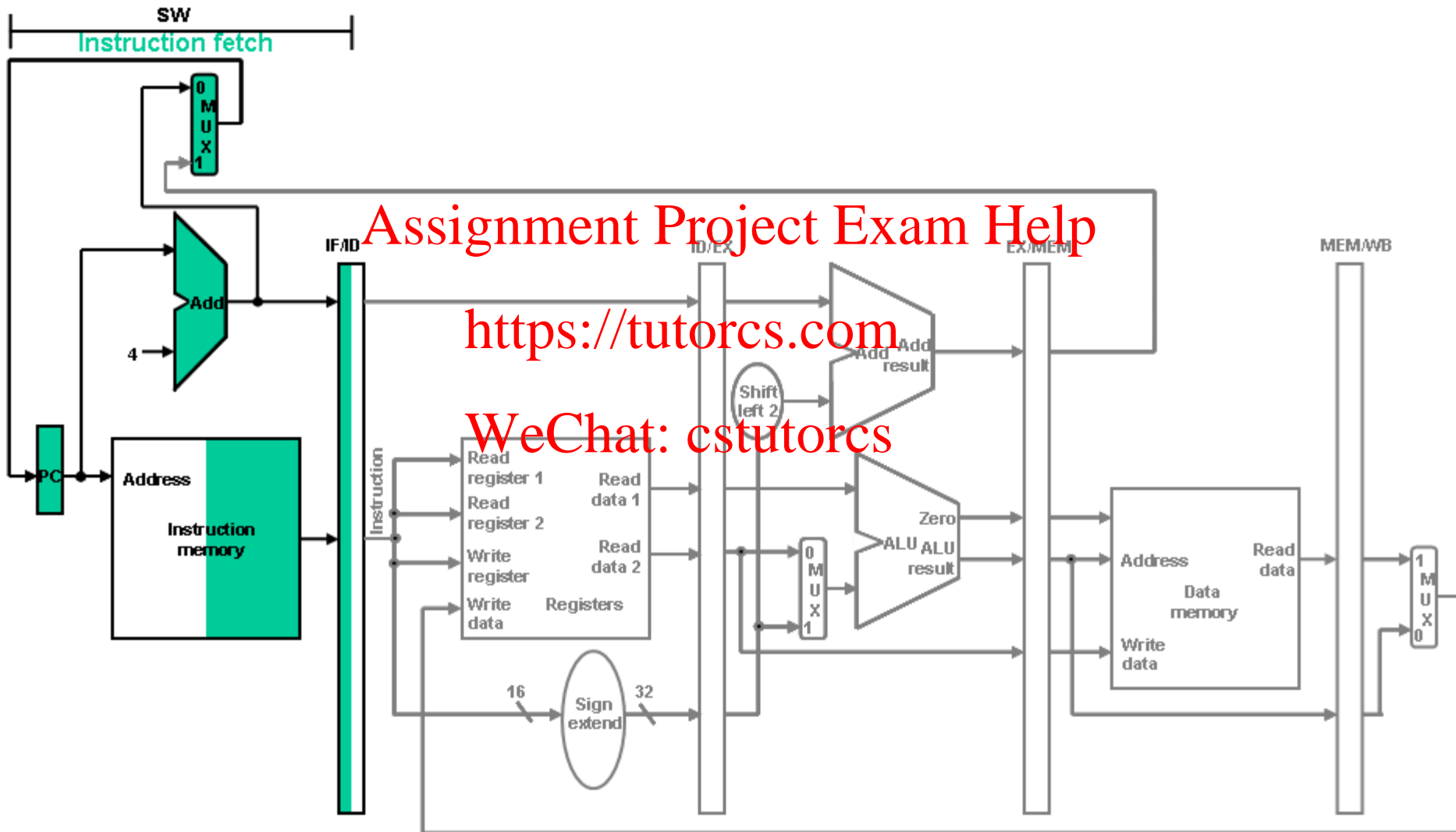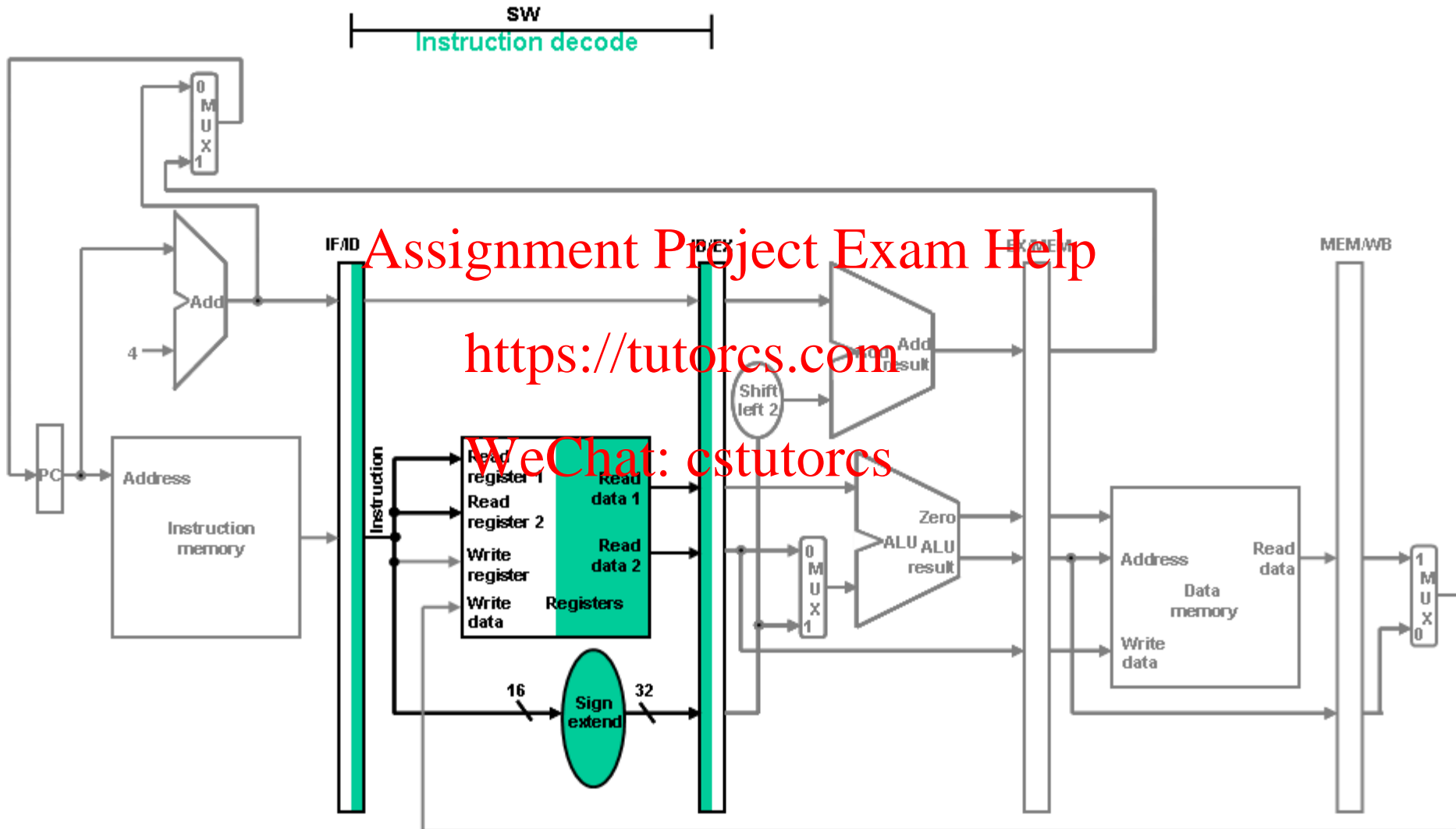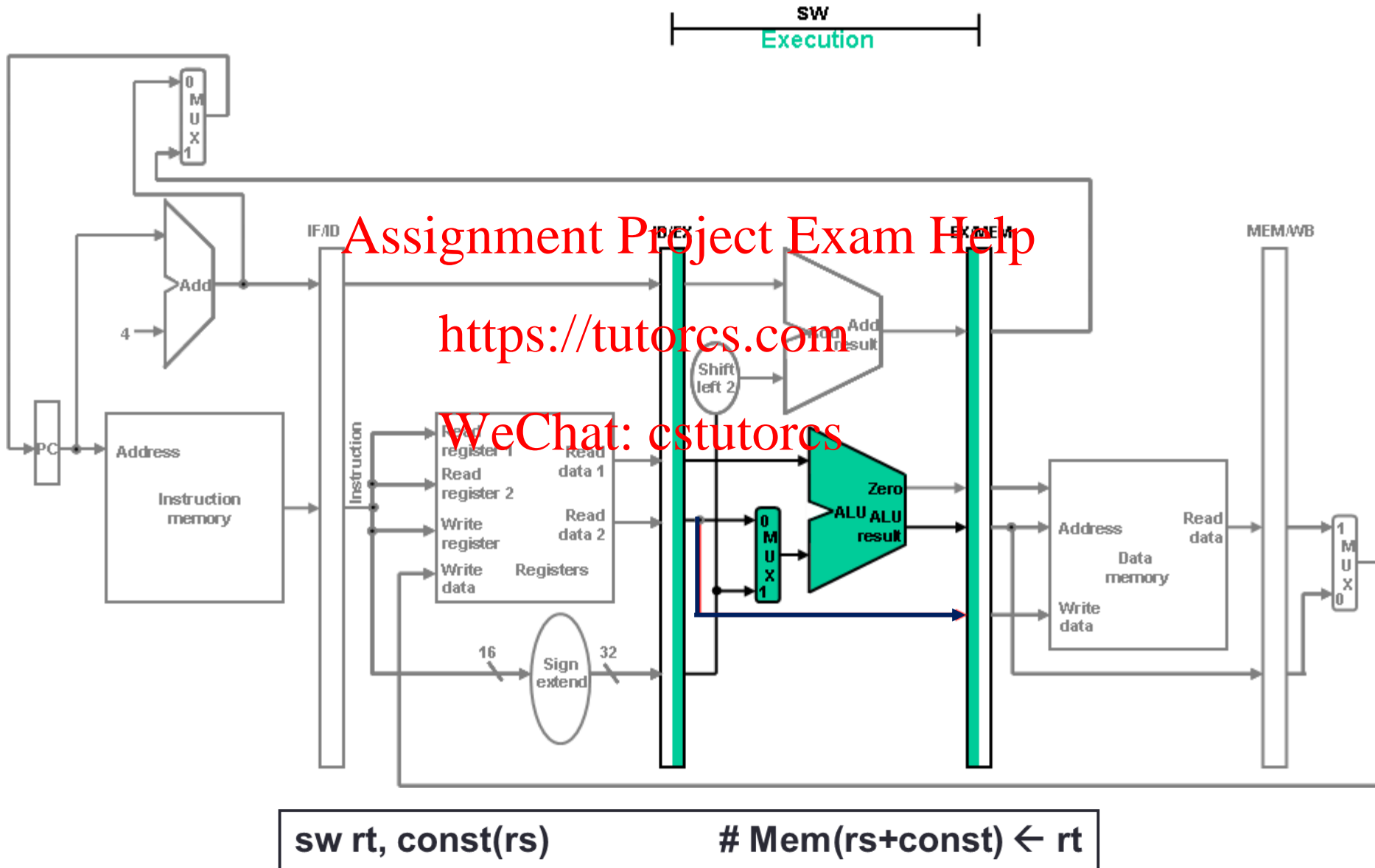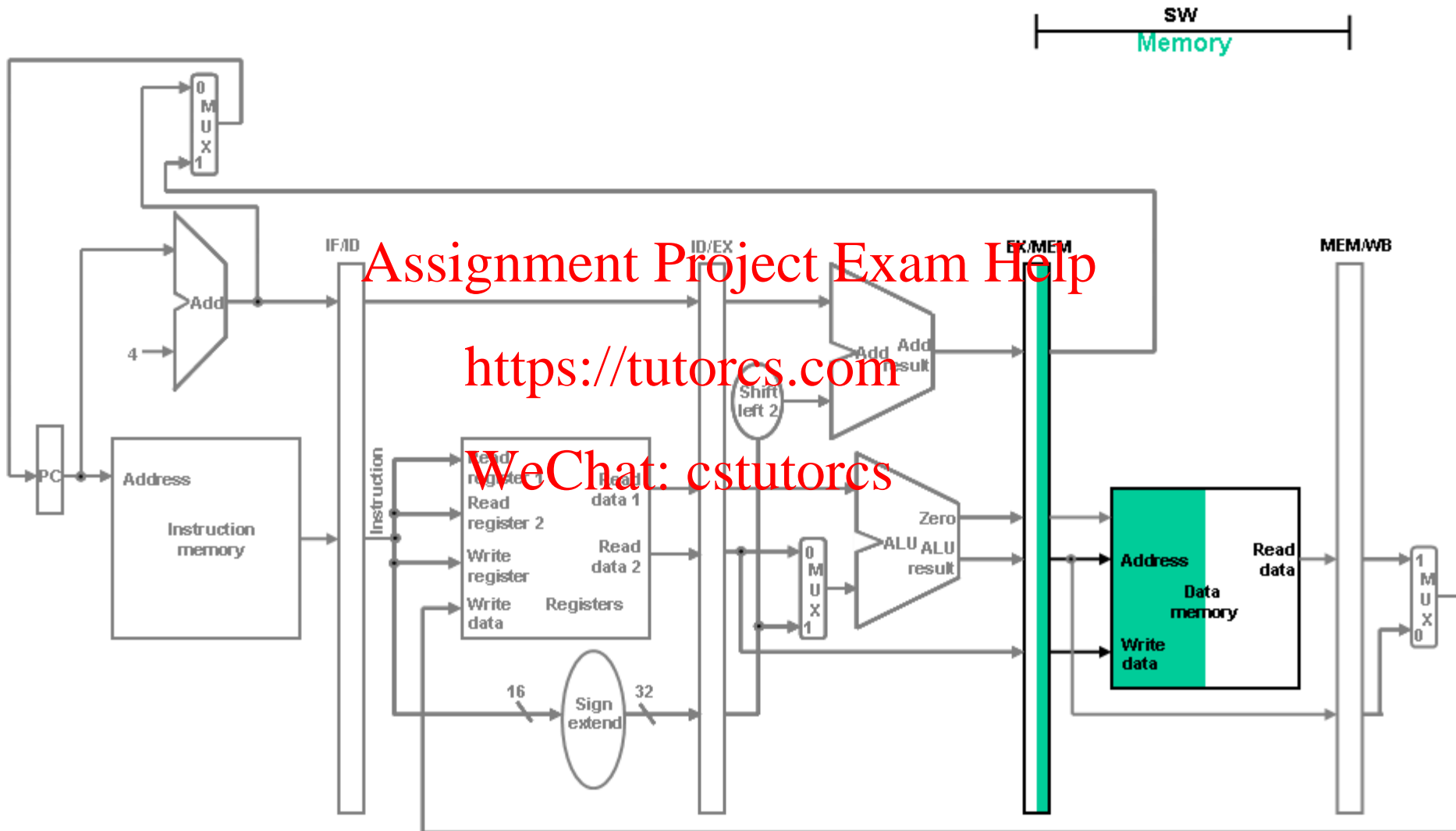
Pipelined SW execution: WB stage

Write back

Assignment Project Exam Help

https://tutorcs.com

WeChat: cstutorcs
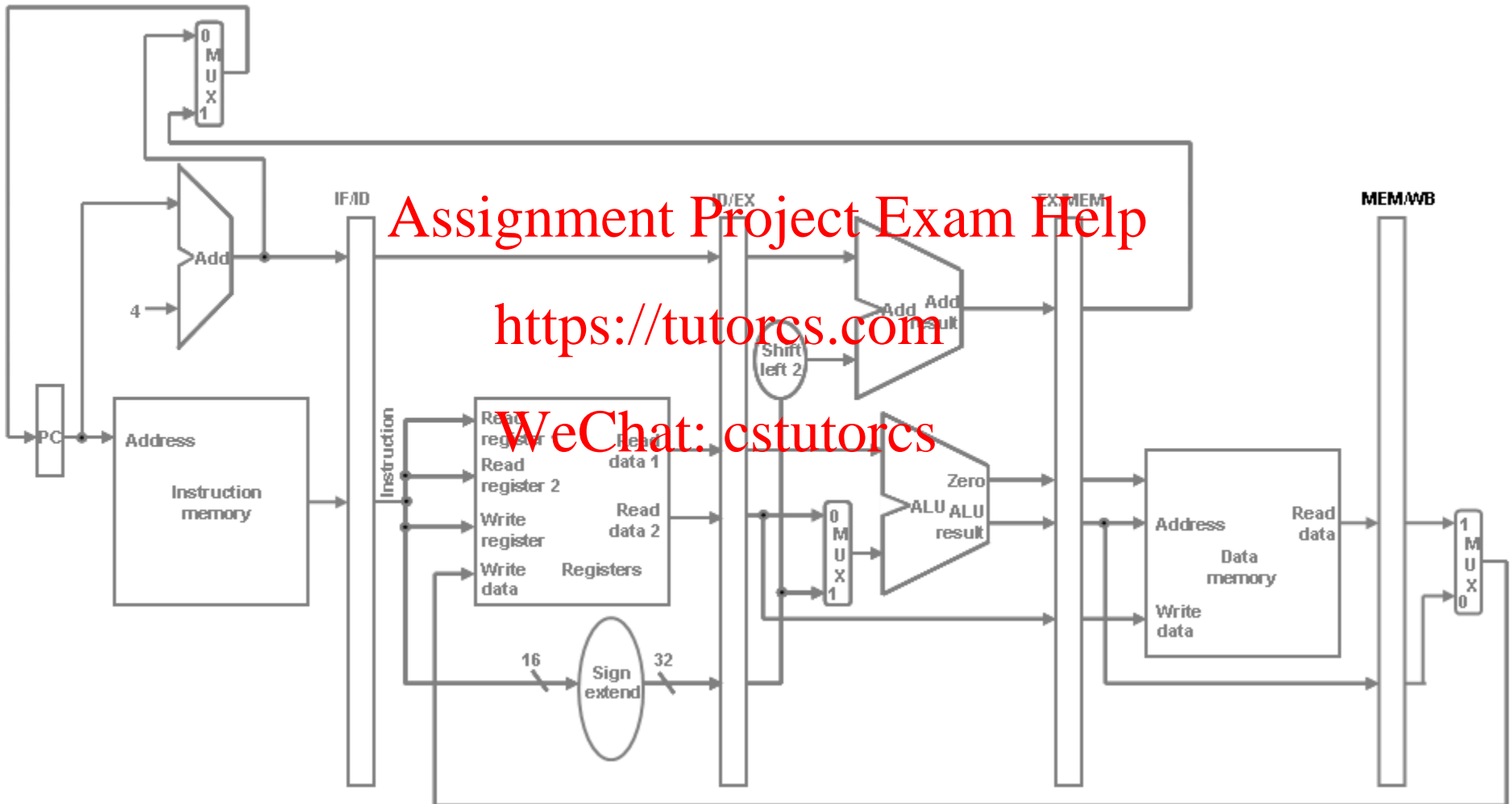
Notes on SW instruction execution

- **The first three stages are identical to those for the load instruction, but the last two stages operate differently**

  - *Memory access*
    - The data is written to memory. In order to make the data available during the Mem stage it had to be placed into the EX/MEM register during the EX stage.

  - *Write back*
    - *F*or this instruction, nothing happens in the write back stage.

Assignment Project Exam Help

https://tutorcs.com

WeChat: cstutorcs

Bug in the write back stage?

Assignment Project Exam Help

https://tutorcs.com

WeChat: cstutorcs
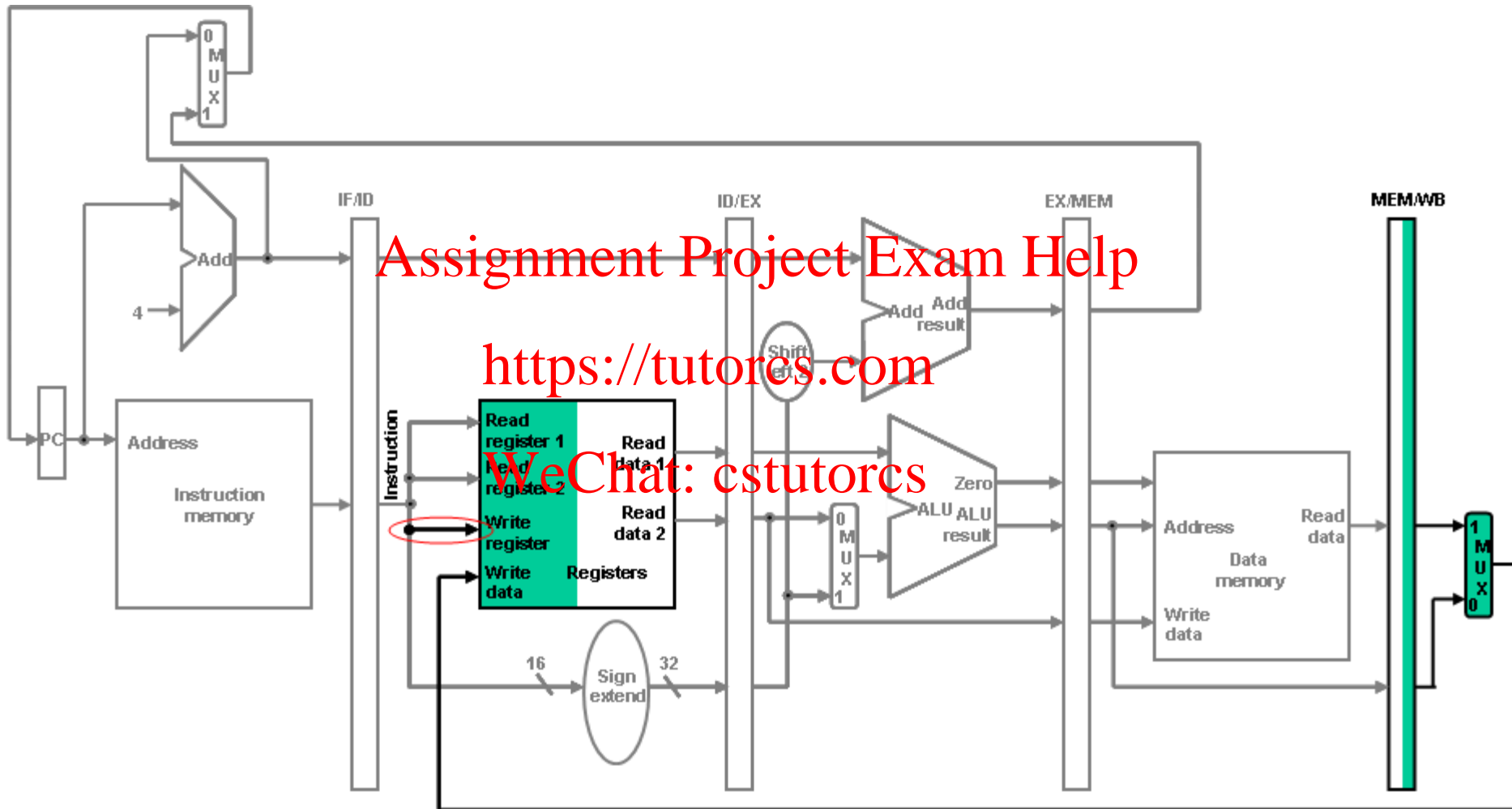
Corrected design

- **Carry the destination register info through the pipeline as well!**

Assignment Project Exam Help

https://tutorcs.com

WeChat: cstutorcs
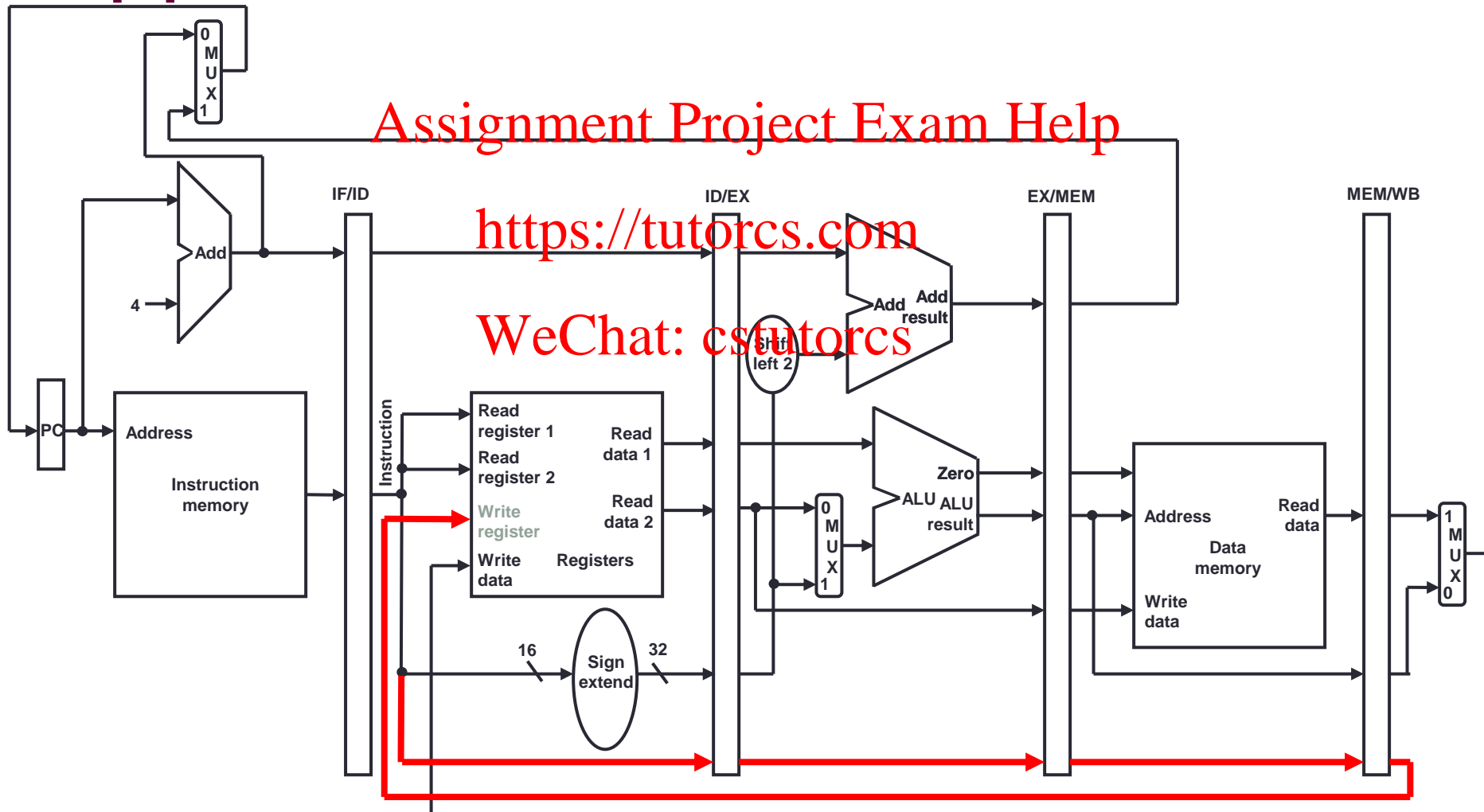
Key points

- **To pass something from an early pipe stage to a later pipe stage, the information must be placed in the pipeline register; otherwise the information will be lost when the next instruction enters that pipeline stage.**

- **Each logical component of the datapath – such as instruction memory, register read ports, ALU, data memory, and register write port – can be used only within a *single* pipeline stage; otherwise we would have a *structural hazard.***
  - **To be covered later**

Pipelined control

- **Add control to the pipelined datapath just like we did to the single cycle datapath:**

  1. **Label control points**

  2. **Determine the control settings for each stage for all instructions**

  3. **Design control logic to implement the control**

     - **To handle sequencing, control inputs are pipelined together with data and intermediate results**

Assignment Project Exam Help

https://tutorcs.com

WeChat: cstutorcs

Control points in the pipelined datapath

Assignment Project Exam Help

https://tutorcs.com

WeChat: cstutorcs

IF/ID   **ID/EX**   **EX/MEM**   **MEM/WB**

0 M U X 1

Add

4

Add Add result

Branch

Shift left 2

RegWrite

PC

Address

Instruction memory

Instruction

Read register 1

Read register 2

Write register

Write data

Registers

Read data 1

Read data 2

ALUSrc

Zero

ALU ALU result

0 M U X 1

MemWrite

Address Read data

Data memory

Write data

MemToReg

1 M U X 0

MemRead

Instruction [15 - 0]   16

Sign extend

32

6

ALU control

ALUOp

Instruction [20 - 16]

0 M U X 1

Instruction [15 - 11]

RegDst

Notes on pipeline control

- **Since each control signal is associated with a component that is active only in one pipeline stage, we group control signals for stages**
  - **IF stage**
    - **No control signals needed**

    Assignment Project Exam Help

  - **ID stage**
    - **ExtOp**

    https://tutorcs.com

  - **Ex stage**

    WeChat: cstutorcs
    - **RegDst, ALUOp, and ALUSrc**
  - **Mem stage**
    - **Branch, MemRead, and MemWrite**
  - **Write back**
    - **MemtoReg and RegWrite**

Recall: Control Unit of single cycle processor

- **We can use the same design for pipelined datapath**

Assignment Project Exam Help

https://tutorcs.com
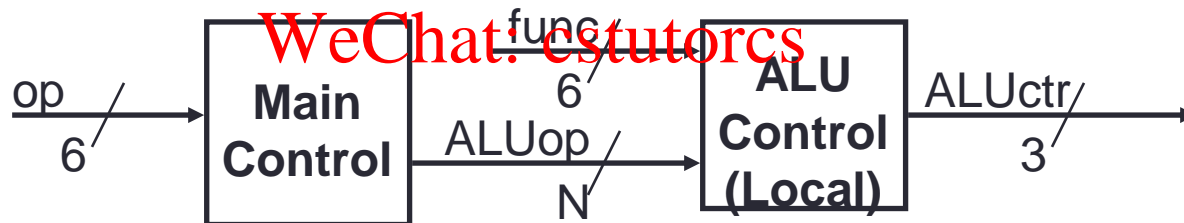
WeChat: cstutorcs

Pipelining the control signals

- **The control signals are generated in ID stage**
- **These control signals are then used in the following stages as the instruction moves down the pipeline**

Assignment Project Exam Help

https://tutorcs.com
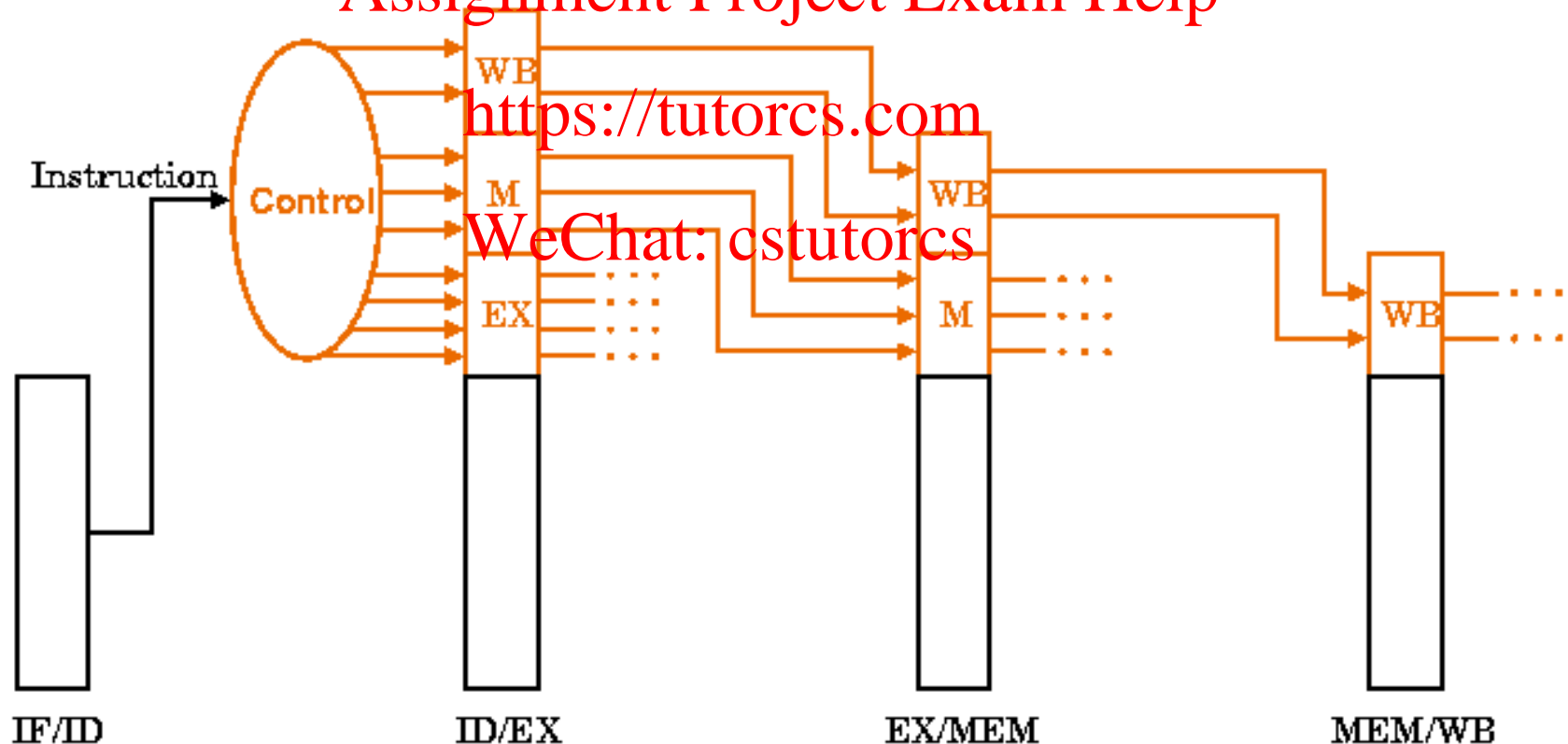
WeChat: cstutorcs

Putting it all together: pipelined processor

Assignment Project Exam Help

https://tutorcs.com
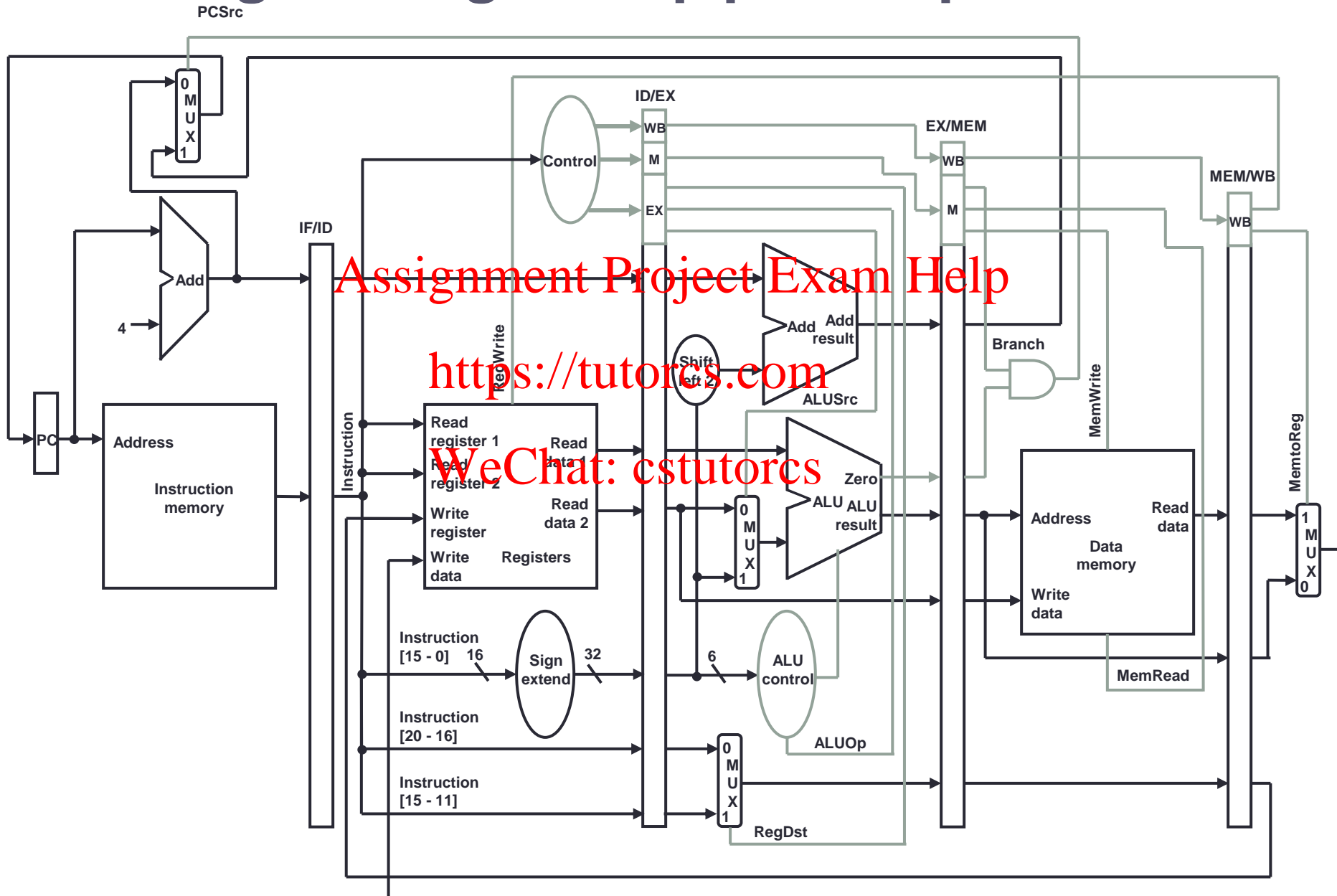
WeChat: cstutorcs

Example of pipeline execution

- **How are the following instructions executed?**

```
lw   $10, 20($1)
sub  $11, $2, $3
and  $12, $4, $5
or   $13, $6, $7
add  $14, $8, $9
```

Execution example: clock cycle 1/9

IF: lw $10, 20($1)   **ID: before <1>**   **EX: before <2>**   **MEM: before <3>**   **WB: b <4>**

Assignment Project Exam Help

https://tutorcs.com

WeChat: cstutorcs

```
lw   $10, 20($1)
sub  $11, $2, $3
and  $12, $4, $5
or   $13, $6, $7
add  $14, $8, $9
```

Execution example: clock cy

IF: sub $11, $2, $3   ID: lw $10, 20($1)   EX: before <1>

- **IF stage and ID stage**
  - No control signals needed.
- **Ex stage**
  - RegDst, ALUOp, and ALUSrc
- **Mem stage**
  - Branch, MemRead, and MemWrite
- **Write back**
  - MemtoReg and RegWrite

PCSrc

ID/EX

IF/ID

Add

4

PC

Address

Instruction memory

Control

lw

11
WB   00
010
M   000
0001
EX   0
00
0

Instruction

1
Read register 1   Read data 1 $1
X
Read register 2   Read data 2 $X
Write register
Write data   Registers

RegWrite

Sign extend

Instruction [15 - 0]   20
20

Instruction [20 - 16]   rt
10   10

Instruction [15 - 11]   rd
X   X

Shift left 2

Add   Add result

ALUSrc

0 MUX 1

ALU   ALU result

Zero

ALU control

ALUOp

0 MUX 1

RegDst

Branch

MemWrite

Address   Read data

Data memory

Write data

MemRead

WB 0

1 MUX 0

MemtoReg

lw $10, 20($1)
→ **sub $11, $2, $3**
  and $12, $4, $5
  or $13, $6, $7
  add $14, $8, $9

Execution example: clock cycle 3/9

IF: and $12, $4, $5   **ID: sub $11, $2, $3**   **EX: lw $10, 20($1)**   **MEM: before <1>**   **WB: b <2>**

PCSrc

0 MUX 1

sub

ID/EX

10   11
WB
Control
000   010
M

1100   00
EX

EX/MEM

00
WB

0

M

MEM/WB

0
WB

IF/ID

Add

4

Assignment Project Exam Help

Add   Add result

https://tutorcs.com

Shift left 2

Branch

ALUSrc

MemWrite

PC

Address

Instruction memory

Instruction

2
Read register 1

3
Read register 2

Write register

Write data

Registers

Read data 1 $2

Read data 2 $3

$1

$X

0 MUX 1

ALU   ALU result

Zero

$X

Address

Data memory

Read data

Write data

Read data

1 MUX 0

MemtoReg

WeChat: cstutorcs

Instruction [15 - 0]

Sign extend

X   20

ALU control

MemRead

lw $10, 20($1)

sub $11, $2, $3

→ and $12, $4, $5

or $13, $6, $7

add $14, $8, $9

X

X

Instruction [20 - 16]

X   10

0 MUX 1

ALUOp

11
Instruction [15 - 11]

11   X

0 MUX 1

10

RegDst

Execution example: clock cycle 4/9

IF: or $13, $6, $7　　　**ID: and $12, $4, $5**　　　**EX: sub $11, $2, $3**　　　**MEM: lw $10, 20($1)**　　　**WB: b <1>**

PCSrc

0 MUX 1

ID/EX

10
WB 10
000
M 000

and

Control

1100
EX 10
0

IF/ID

Add

4

Add result

Branch

Shift left 2

EX/MEM

WB 11
0
M 1
0

MEM/WB

WB 0
0

Assignment Project Exam Help

https://tutorcs.com

WeChat: cstutorcs

PC

Address

Instruction memory

Instruction

4 Read register 1

5 Read register 2

Write register

Write data

Registers

RegWrite

Read data 1 $4

Read data 2 $5

$2

$3

ALUSrc

0 MUX 1

Zero

ALU

ALU result

$3

Address

Data memory

Read data

Write data

MemWrite

MemRead

MemtoReg

1 MUX 0

Instruction [15 - 0]

Sign extend

X X X

ALU control

ALUOp

lw $10, 20($1)
sub $11, $2, $3
and $12, $4, $5
→ or $13, $6, $7
add $14, $8, $9

Instruction [20 - 16]

X X X

0 MUX 1

Instruction [15 - 11]

12 12 11

RegDst

11 10

Execution example: clock cycle 5/9

IF: add $14, $8, $9   **ID: or $13, $6, $7**   **EX: and $12, $4, $5**   **MEM: sub $11, $2, $3**   **WB: lw…**

PCSrc

0 MUX 1

or

ID/EX

10

10

WB

Control

000

000

M

1100

1

10

EX

0

IF/ID

Add

Assignment Project Exam Help

Add result

4

https://tutores.com

Branch

Shift left 2

WeChat: cstutorcs

ALUSrc

RegWrite

6

Read register 1

Read data 1 $6

$4

7

Read register 2

$7

$5

Zero

Instruction

10

Write register

Read data 2

0 MUX 1

ALU ALU result

Address

PC

Address

Instruction memory

Write data

Registers

$5

Address

Data memory

Read data

1 MUX 0

Write data

X

Instruction [15 - 0]

Sign extend

X

X

ALU control

MemRead

lw $10, 20($1)

X

Instruction [20 - 16]

X

X

ALUOp

sub $11, $2, $3

0 MUX 1

and $12, $4, $5

12

11

10

or $13, $6, $7

13

Instruction [15 - 11]

13

12

→ add $14, $8, $9

RegDst

EX/MEM

WB

10

M

0
0
0

MEM/WB

WB

1
1

MemWrite

MemtoReg

Execution example: clock cycle 6/9

IF: after <1>   **ID: add $14, $8, $9**   **EX: or $13, $6, $7**   **MEM: and $12, $4, $5**   **WB: sub…**

PCSrc

0 M U X 1

add

ID/EX

10

10

Control

000

000

10

1100

10

EX

0

EX/MEM

WB

10

WB

0

M

0

MEM/WB

M

0

WB

1

0

Add

4

Assignment Project Exam Help

Add   Add
       result

Branch

https://tutores.com

Shift
left 2

ALUSrc

MemWrite

PC

Address

RegWrite

8

Read
register 1

$8

$6

Read
data 1

9

Read
register 2

$9

$7

Zero

ALU ALU
     result

Address   Read
           data

WeChat: cstutorcs

Instruction
memory

11

Write
register

Read
data 2

0 M U X 1

Data
memory

MemtoReg

Write
data   Registers

$7

Write
data

Instruction
[15 - 0]

X

X

X

Sign
extend

ALU
control

1 M U X 0

lw $10, 20($1)
sub $11, $2, $3
and $12, $4, $5
or $13, $6, $7
add $14, $8, $9

MemRead

Instruction
[20 - 16]

X

X

X

0 M U X 1

ALUOp

13

12

11

Instruction
[15 - 11]

14

14

13

RegDst

Execution example: clock cycle 7/9

IF: after <2>   **ID: after <1>**   **EX: add $14, $8, $9**   **MEM: or $13, $6, $7**   **WB: and…**

Assignment Project Exam Help

https://tutorcs.com

WeChat: cstutorcs

```
lw    $10, 20($1)
sub   $11, $2, $3
and   $12, $4, $5
or    $13, $6, $7
add   $14, $8, $9
```

Execution example: clock cycle 8/9

IF: after <3>　　**ID: after <2>**　　**EX: after <1>**　　**MEM: add $14, $8, $9**　　**WB: or…**

Assignment Project Exam Help

https://tutores.com

WeChat: cstutorcs

```
lw    $10, 20($1)
sub   $11, $2, $3
and   $12, $4, $5
or    $13, $6, $7
add   $14, $8, $9
```
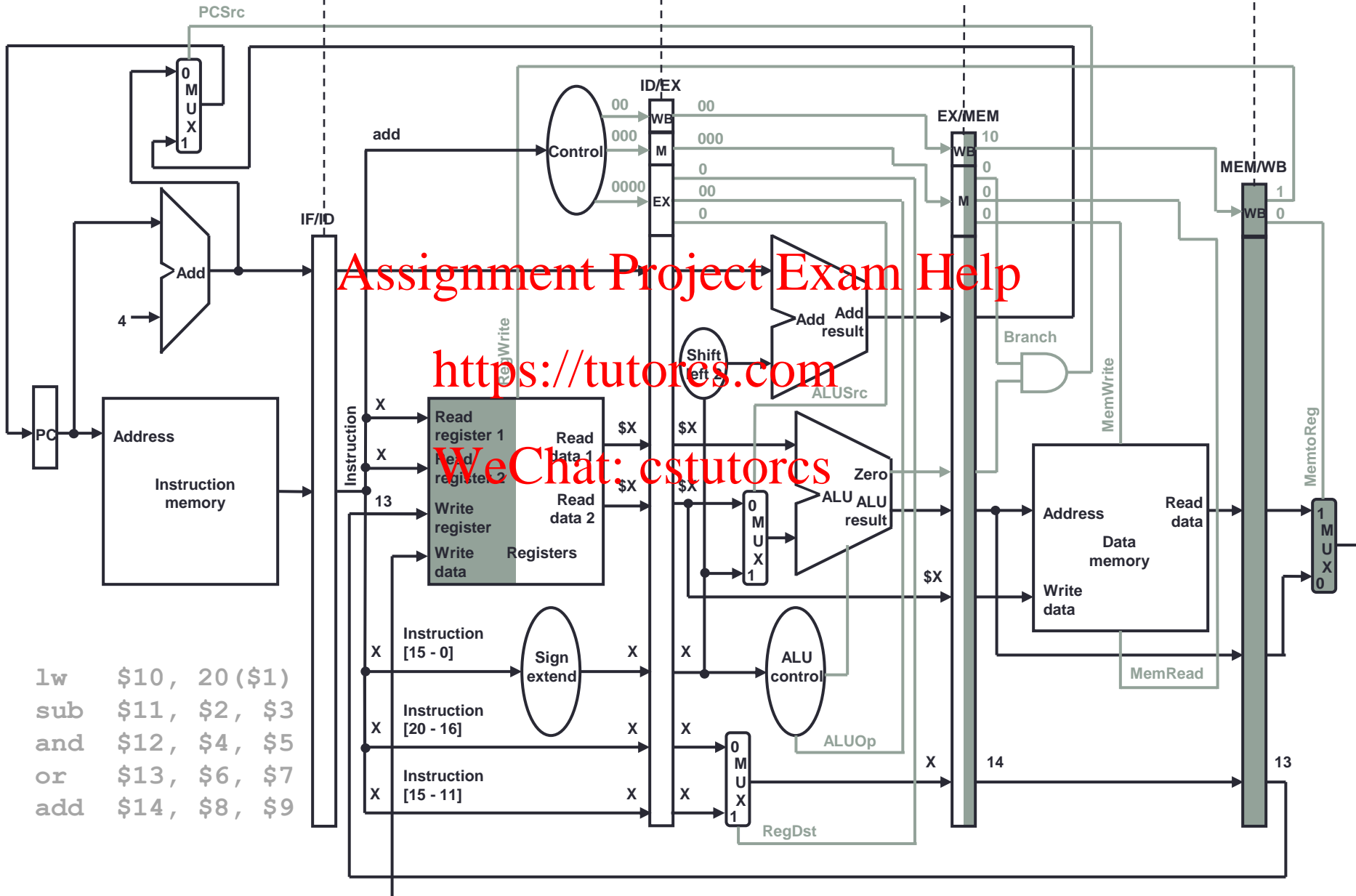
Execution example: clock cycle 9/9

IF: after <4>   **ID: after <3>**   **EX: after <2>**   **MEM: after <1>**   **WB: add…**

PCSrc

0 MUX 1

ID/EX

00

00
WB

Control

000
M

000

0

0000
EX

00

0

EX/MEM

00
WB

0

0
M

0

MEM/WB

1
WB

0

IF/ID

Add

4

Assignment Project Exam Help

https://tutorcs.com

WeChat: cstutorcs

Add   Add result

Shift left 2

Branch

MemWrite

PC

Address

Instruction memory

Instruction

X   Read register 1

X   Read register 2

14   Write register

Write data   Registers

RegWrite

Read data 1

Read data 2

$X

$X

$X

$X

0 MUX 1

ALUSrc

Zero

ALU ALU result

$X

Address

Data memory

Write data

Read data

1 MUX 0

MemtoReg

Instruction [15 - 0]

Sign extend

X

X

ALU control

MemRead

Instruction [20 - 16]

X

X

X

0 MUX 1

ALUOp

X   X

Instruction [15 - 11]

X

X

X

0 MUX 1

RegDst

14

```
lw   $10, 20($1)
sub  $11, $2, $3
and  $12, $4, $5
or   $13, $6, $7
add  $14, $8, $9
```

In-class exercise

Given stage delays as below

- **100ps for register read or write**
- **200ps for other stages**

Compare the performance of the pipelined datapath with

Assignment Project Exam Help

https://tutorcs.com

- **single-cycle datapath**
- **multi-cycle datapath**

WeChat: cstutorcs

Instr	Instr fetch	Register read	ALU op	Memory access	Register write	Total time used
lw	200ps	100 ps	200ps	200ps	100 ps	800ps
sw	200ps	100 ps	200ps	200ps		700ps
R-format	200ps	100 ps	200ps		100 ps	600ps
beq	200ps	100 ps	200ps			500ps

Pipeline speedup

- ## If all stages are balanced
  - ### i.e. equal stage delay

$$\text{speedup} = \frac{\text{exec\_time}_{\text{non-pipe}}}{\text{exec\_time}_{\text{pipe}}}$$
$$= \text{no. of stages}$$

- ## If not balanced, speedup is less

- ## Speedup is due to increased throughput
  - ### Latency (time for each instruction) does not decrease

- ## Any other issues?