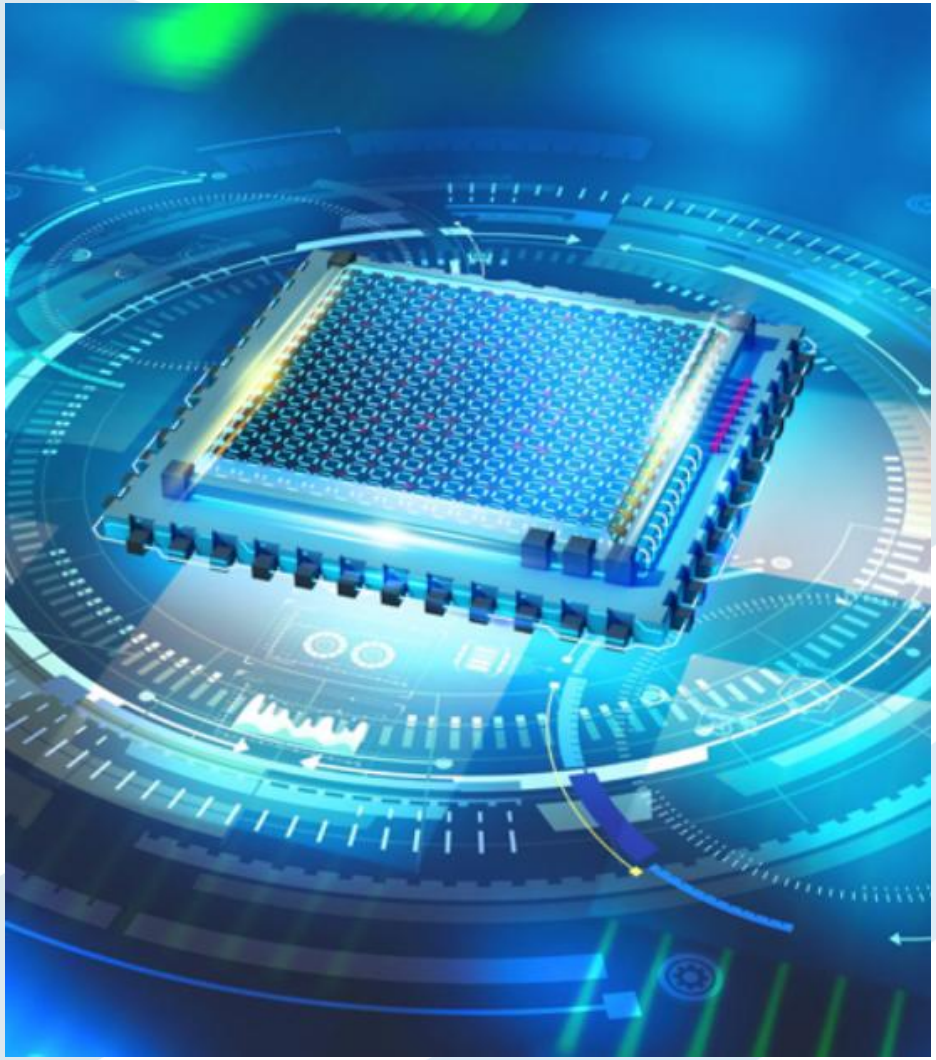
جامعة المنارة الخاصة
كلـــــية الهندســـــة
هندسة ميـــكاترونيك

المعالجات الصغرية ولغة التجميع

المحاضرة التاسعة

مدرس المقرر
د. بسام حسن

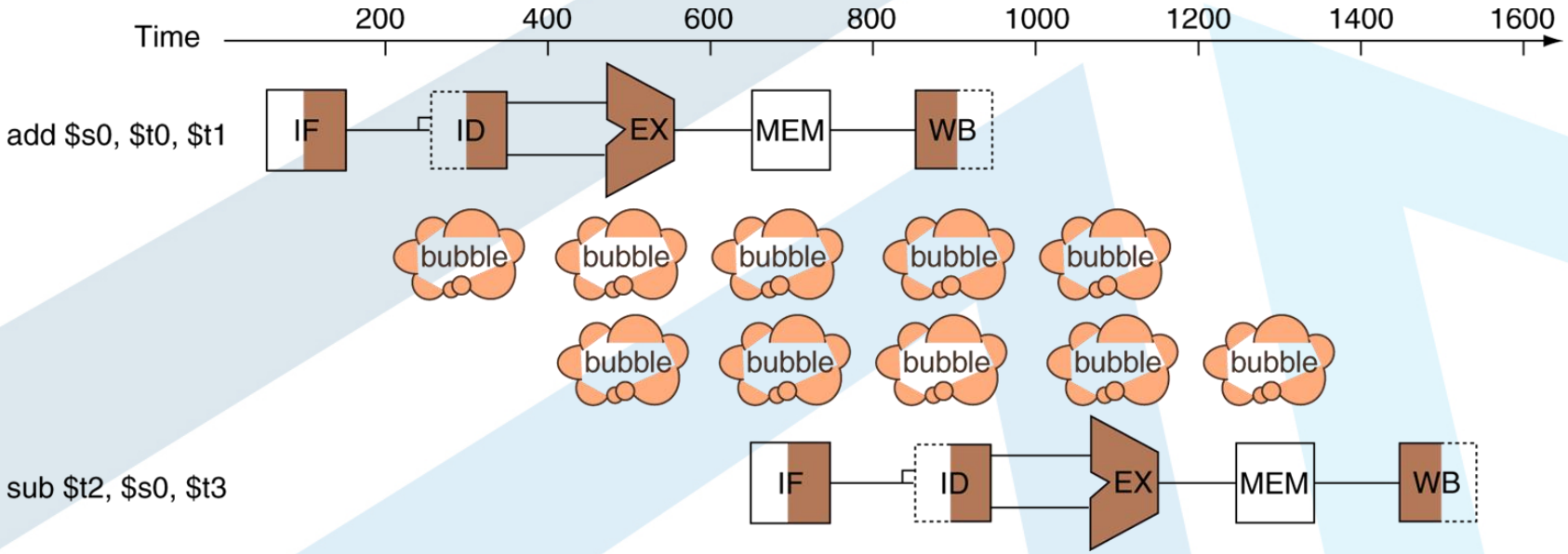
2025_2026



مفردات من المحاضرة التاسعة :

- MIPS Pipelining
- Hazards
- Data Hazards
- Control Hazards

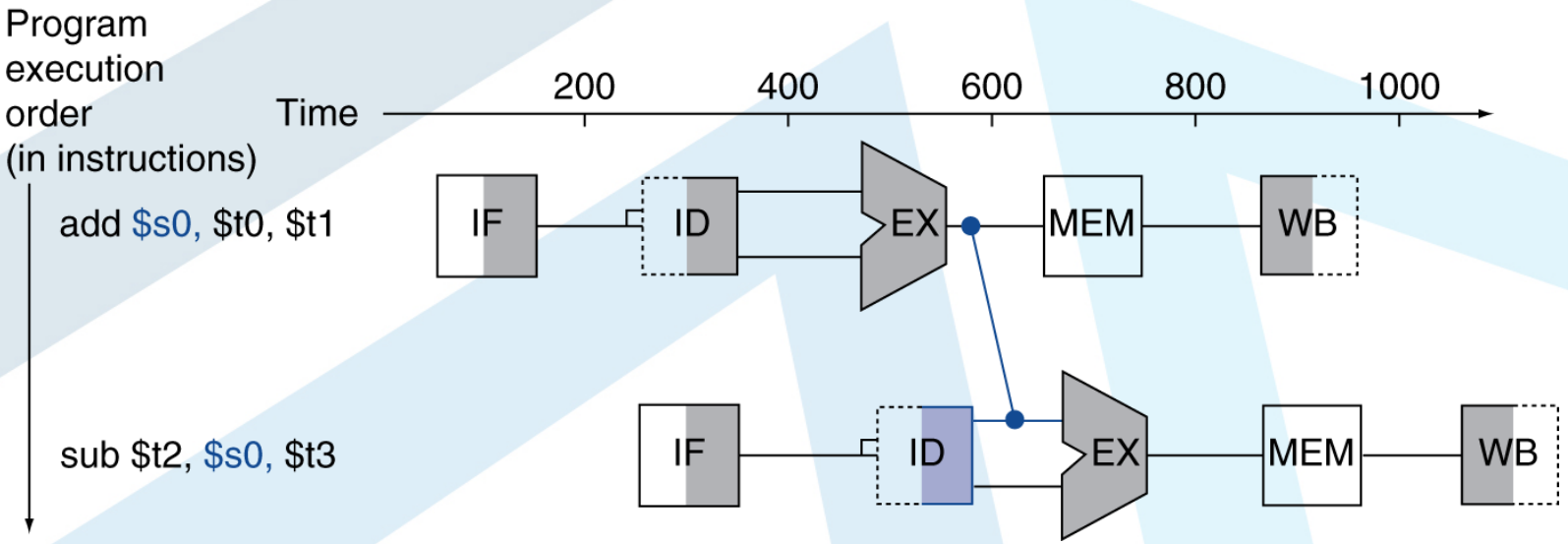
- An instruction depends on completion of data access by a previous instruction
  - `add $s0, $t0, $t1`
    `sub $t2, $s0, $t3`



الحل :

Forwarding (Bypassing)

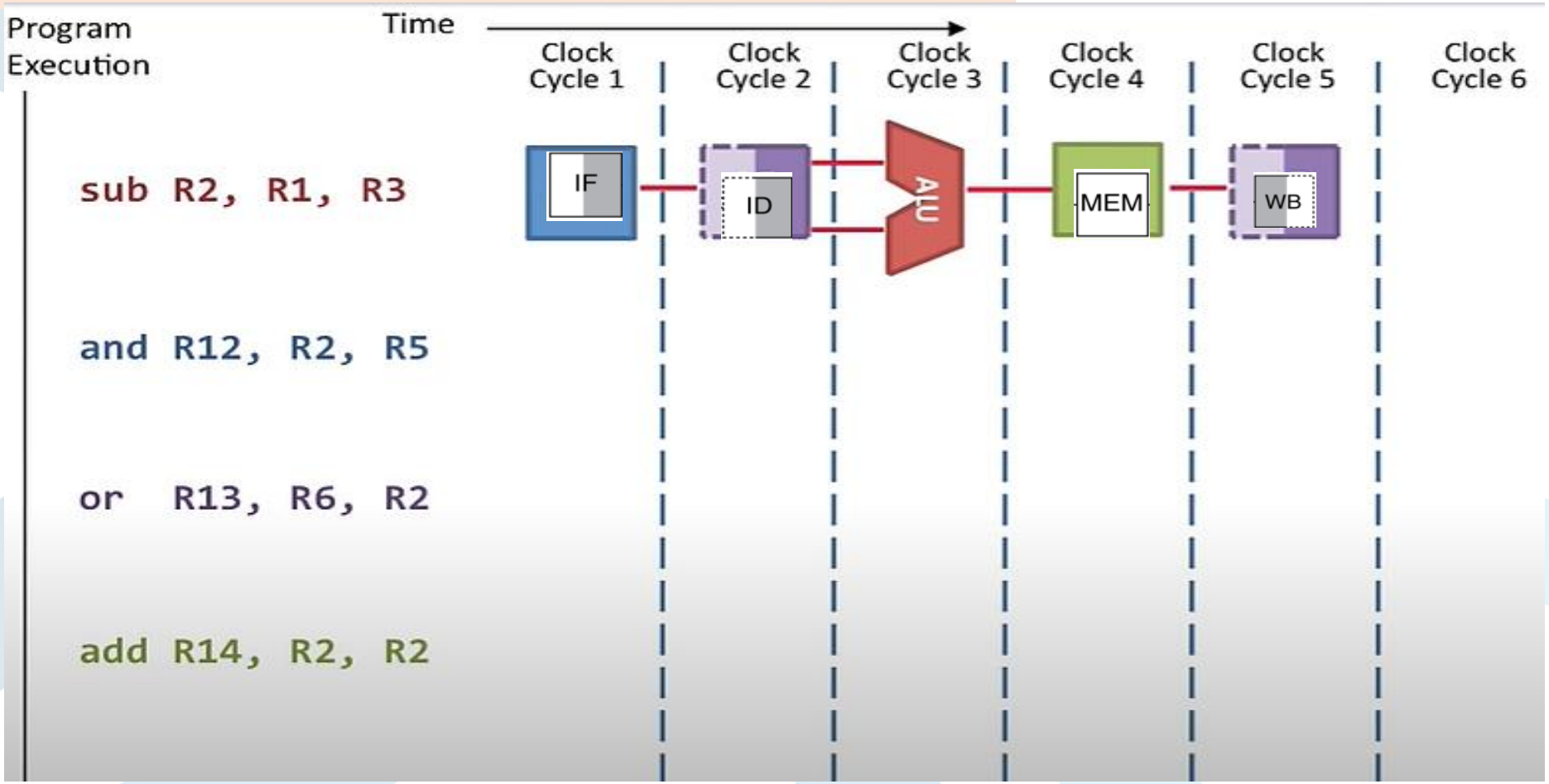
- Use result when it is computed
  - Don’t wait for it to be stored in a register
  - Requires extra connections in the datapath



الحل :

Forwarding (Bypassing)

Where does Forwarding (Bypassing) help?

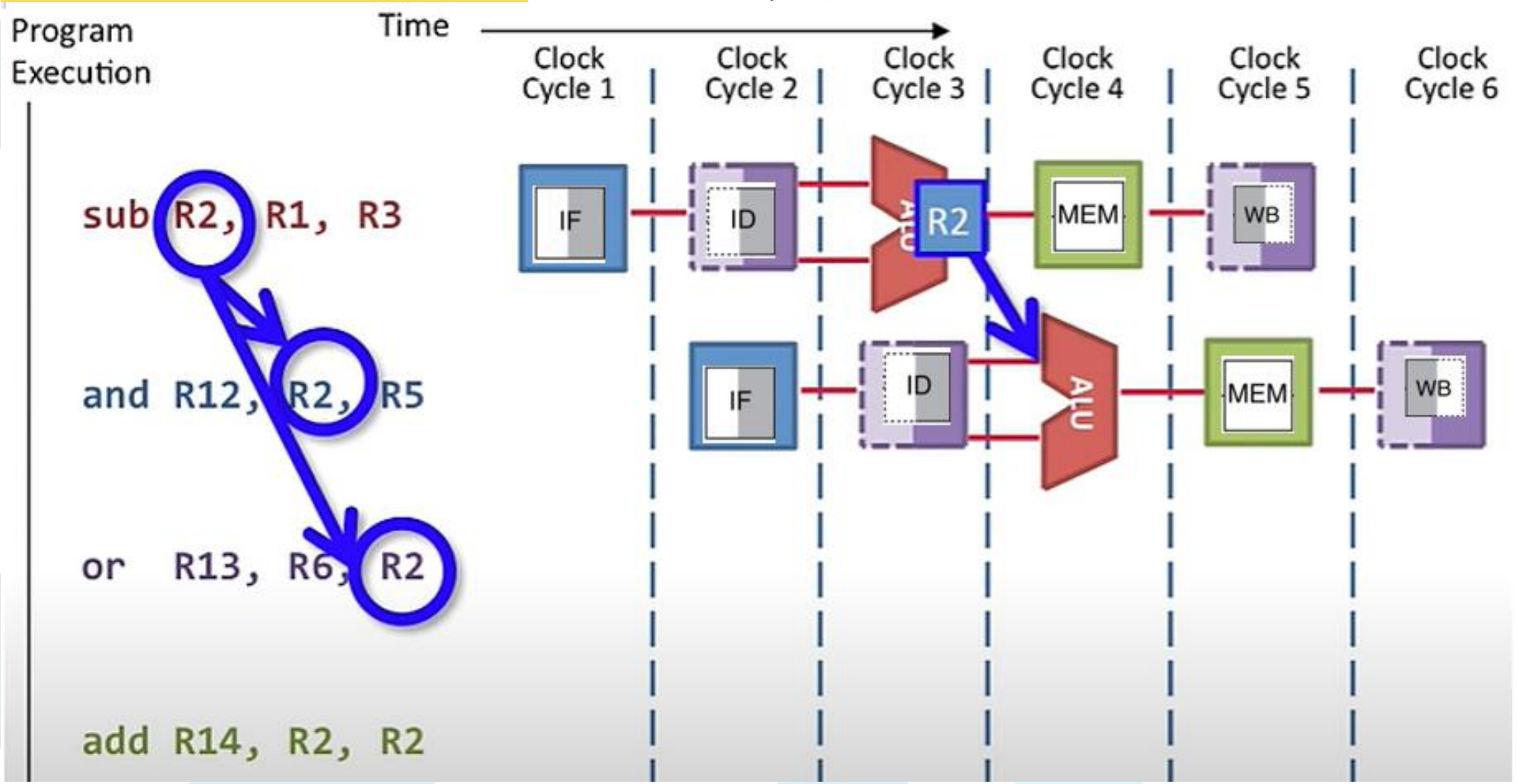


Data Hazards

الحل :

Forwarding (Bypassing)

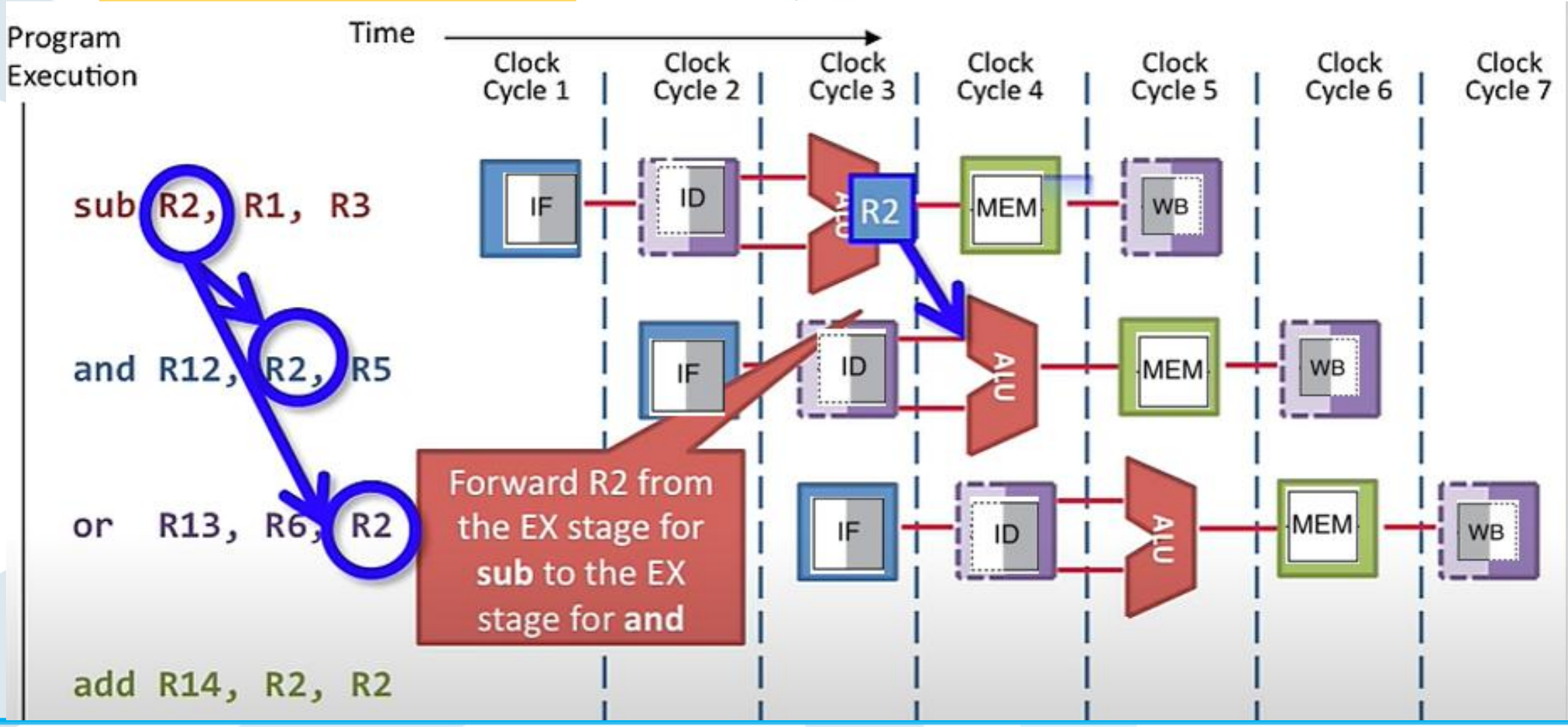
Where does Forwarding (Bypassing) help?



الحل :

Forwarding (Bypassing)

Where does Forwarding (Bypassing) help?

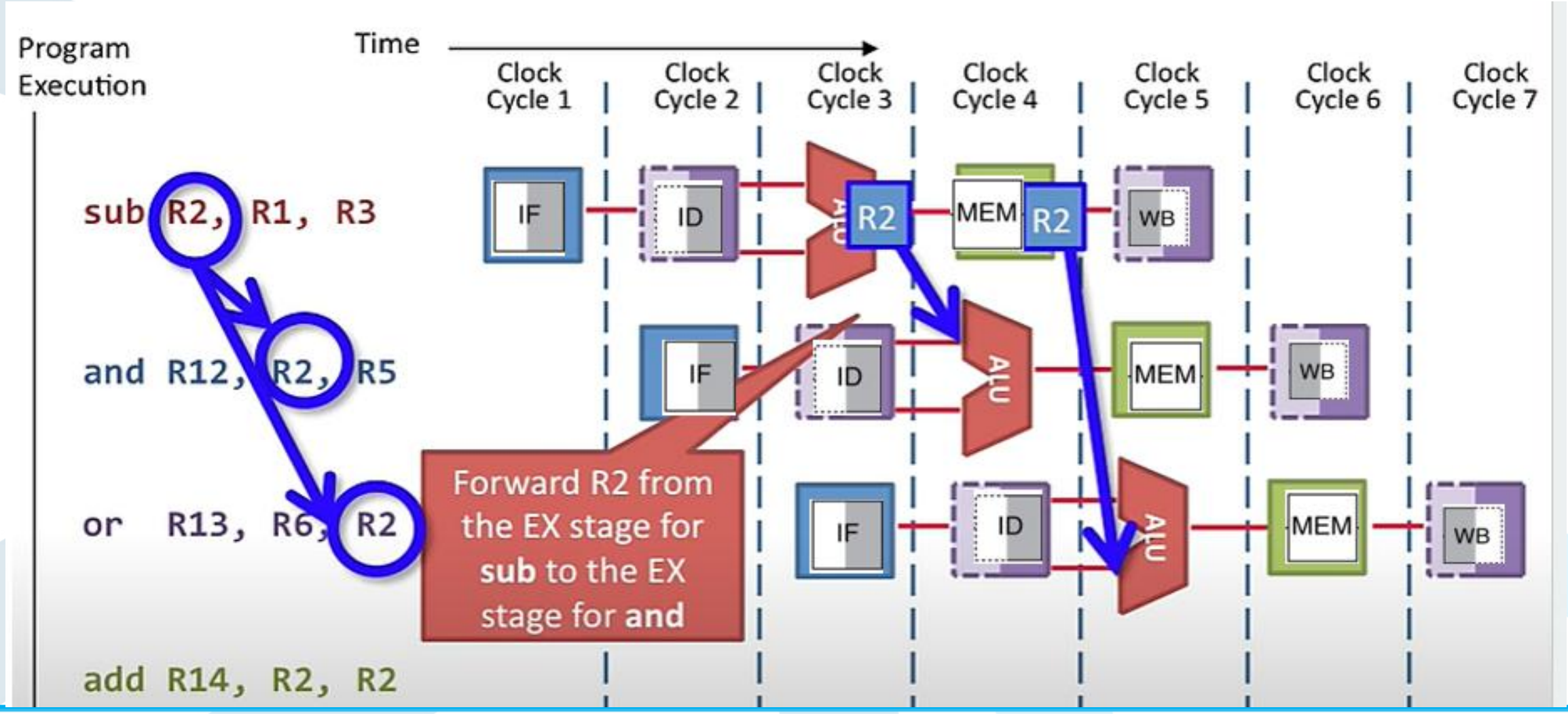


Data Hazards

الحل :

Forwarding (Bypassing)

Where does Forwarding (Bypassing) help?

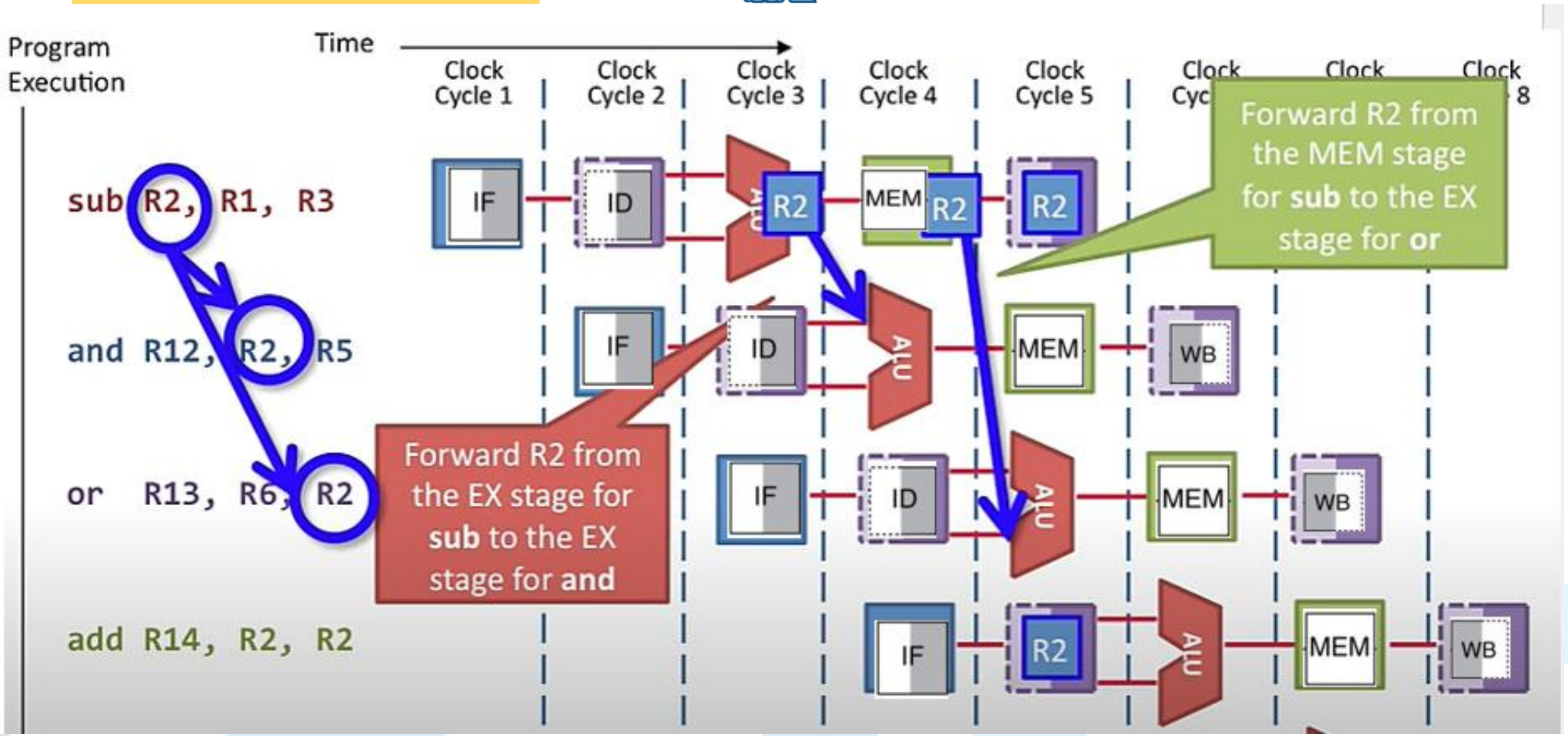


Data Hazards

الحل :

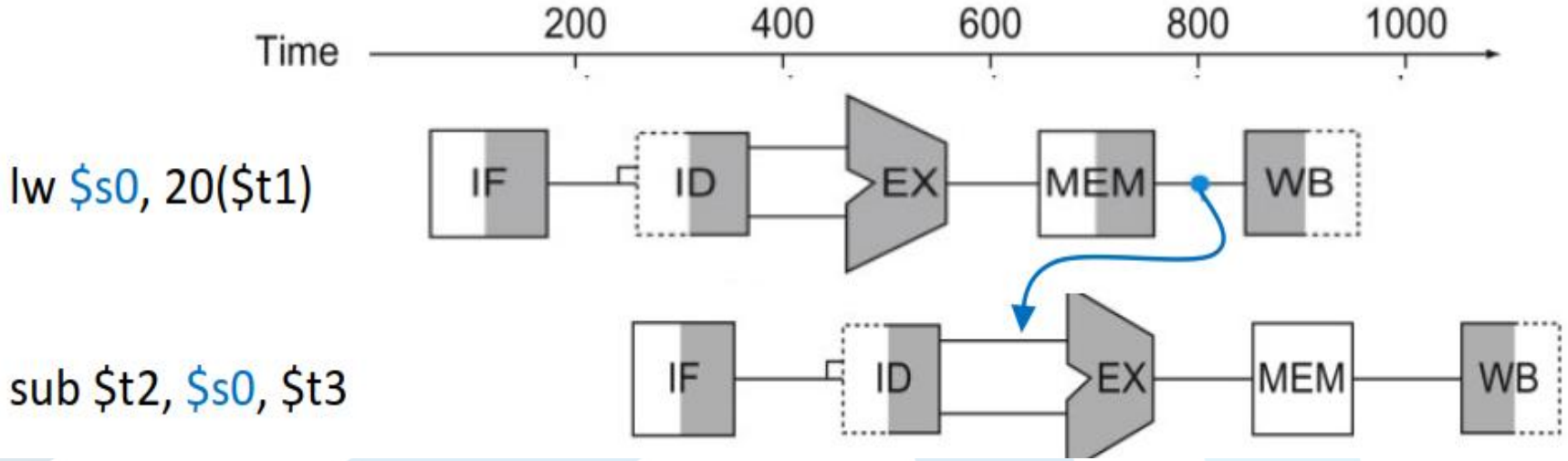
Forwarding (Bypassing)

Where does Forwarding (Bypassing) help?



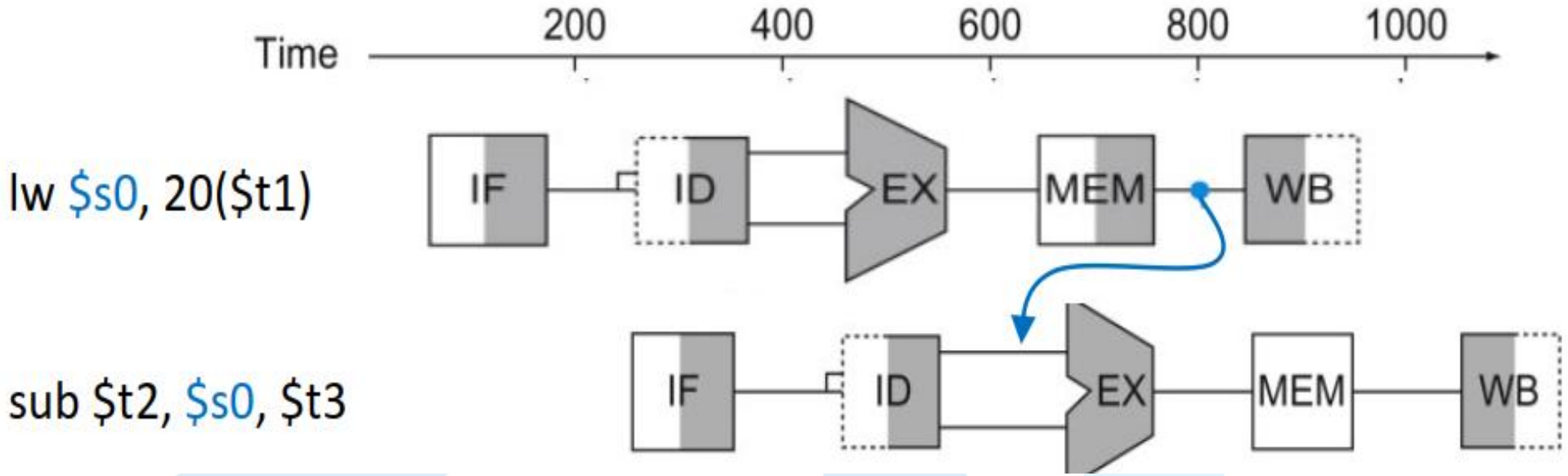
Load-Use Data Hazard

- Can’t always avoid stalls by forwarding
  - If value not computed when needed
  - Can’t forward backward in time!



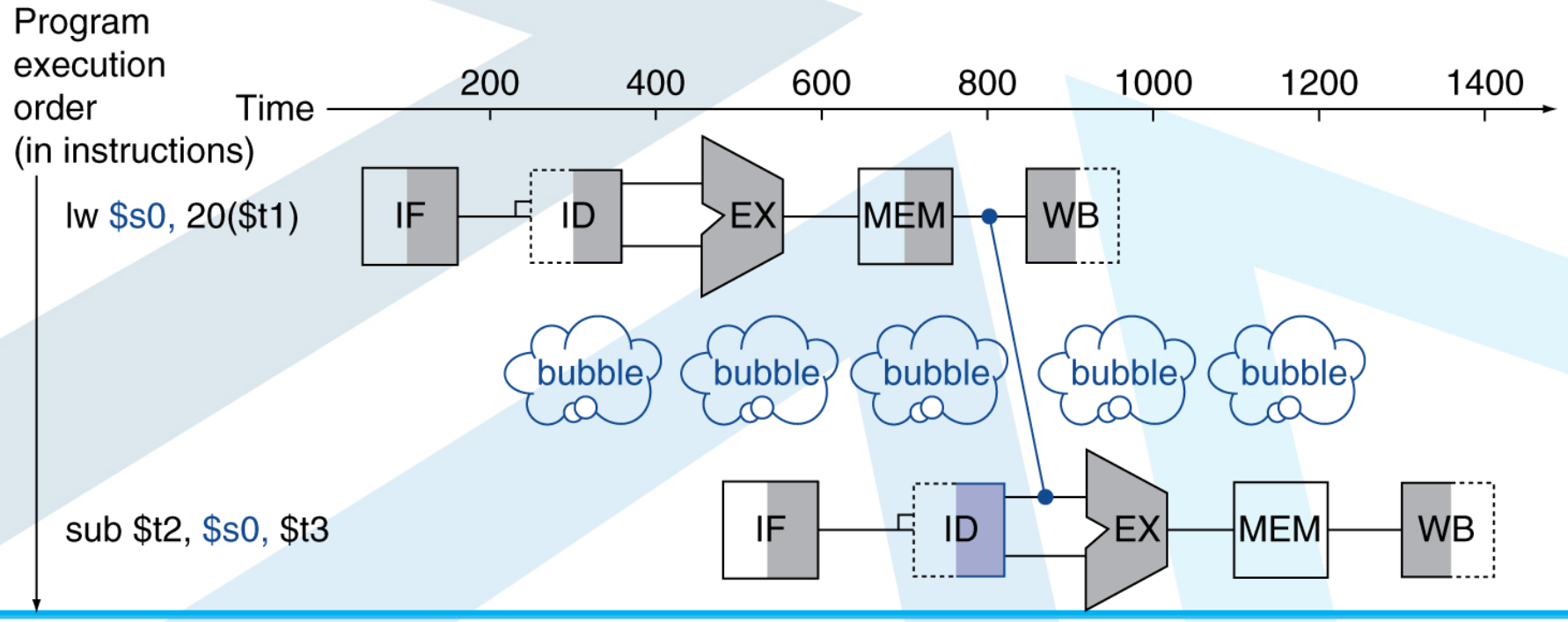
Load-Use Data Hazard

- Can’t always avoid stalls by forwarding
  - If value not computed when needed
  - Can’t forward backward in time!



Load-Use Data Hazard

- Can’t always avoid stalls by forwarding
  - If value not computed when needed
  - Can’t forward backward in time!



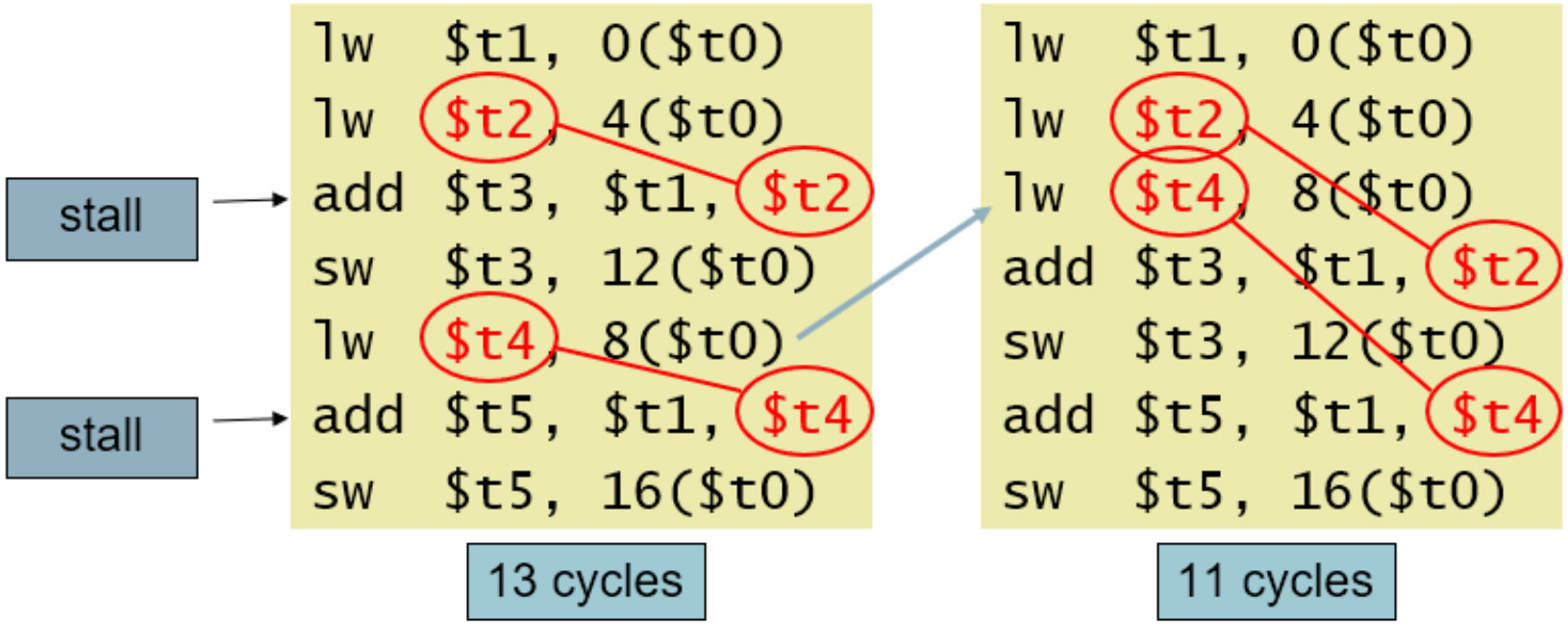
Load-Use Data Hazard

الحل :

Code Scheduling to Avoid Stalls

استخدام الحل البرمجي (software) لإعادة ترتيب الشيفرة لمحاولة تفادي التوقفات الناتجة عن خطر استخدام التعليمة Load.

- Reorder code to avoid use of load result in the next instruction
- C code for A = B + E; C = B + F;



Load-Use Data Hazard

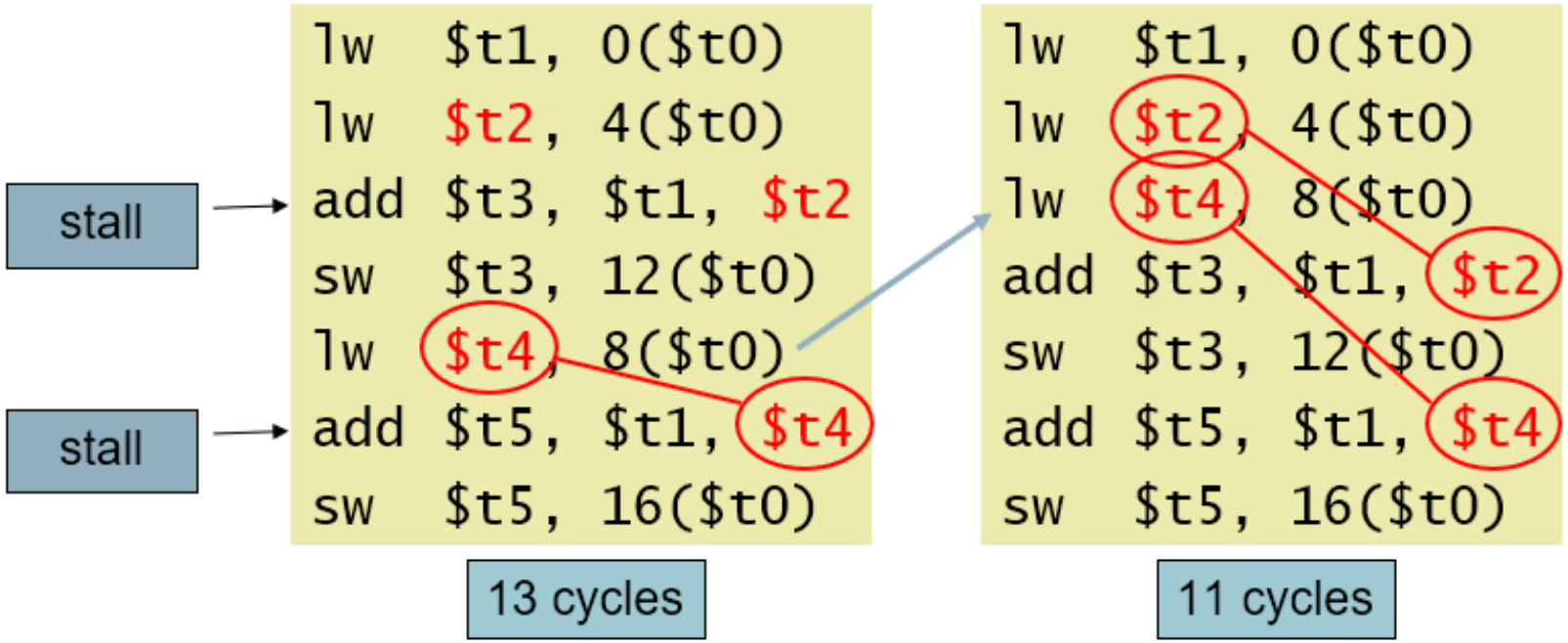
جَـامعة المَـنارة

الحل :

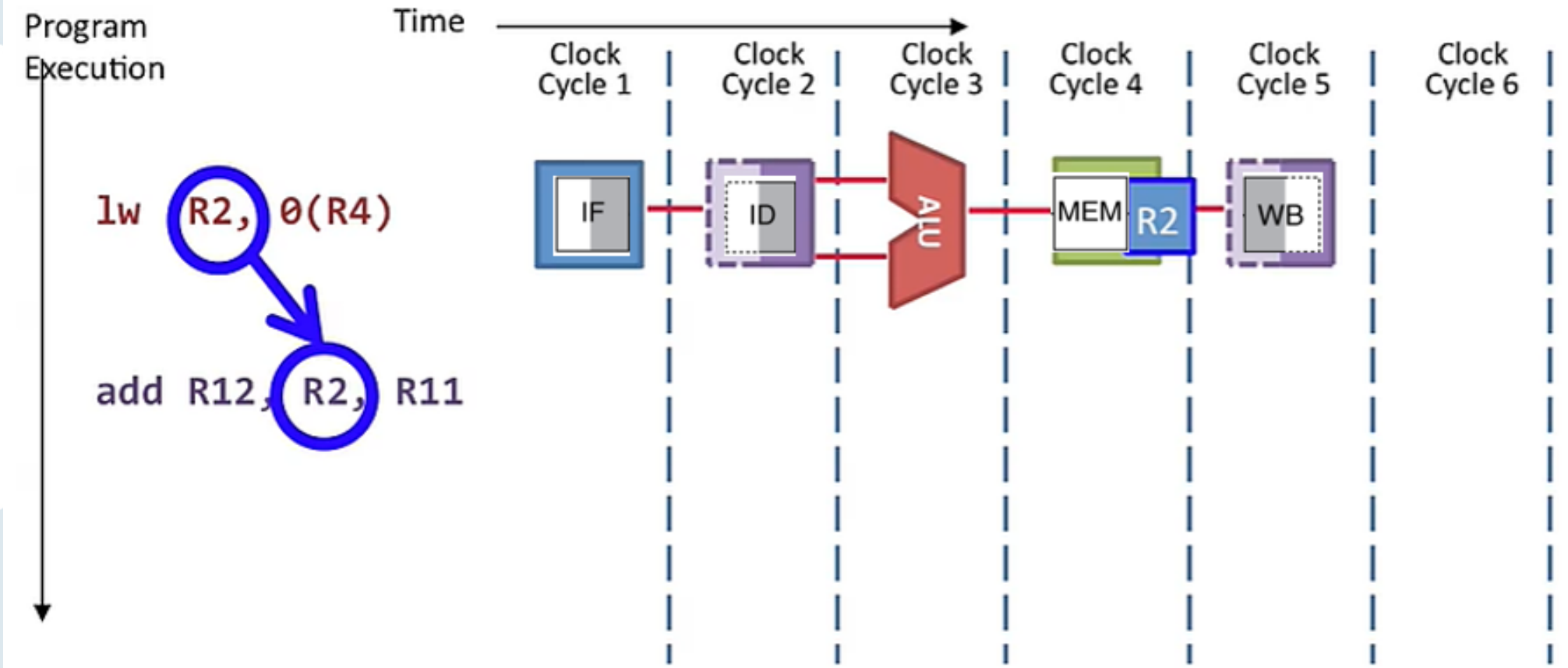
Code Scheduling to Avoid Stalls

استخدام الحل البرمجي (software) لإعادة ترتيب الشيفرة لمحاولة تفادي التوقفات الناتجة عن خطر استخدام التعليمة Load.

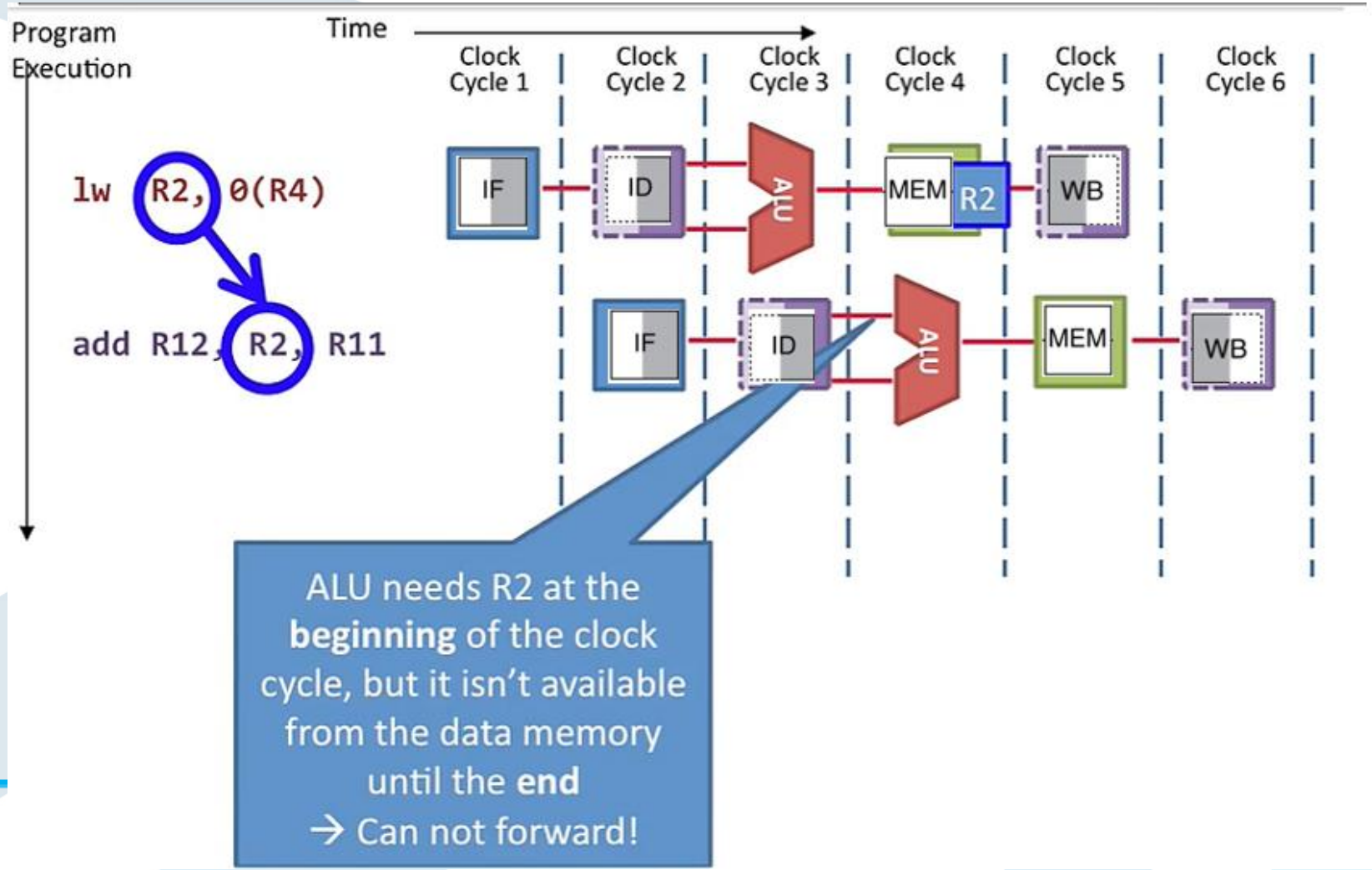
- Reorder code to avoid use of load result in the next instruction
- C code for A = B + E; C = B + F;



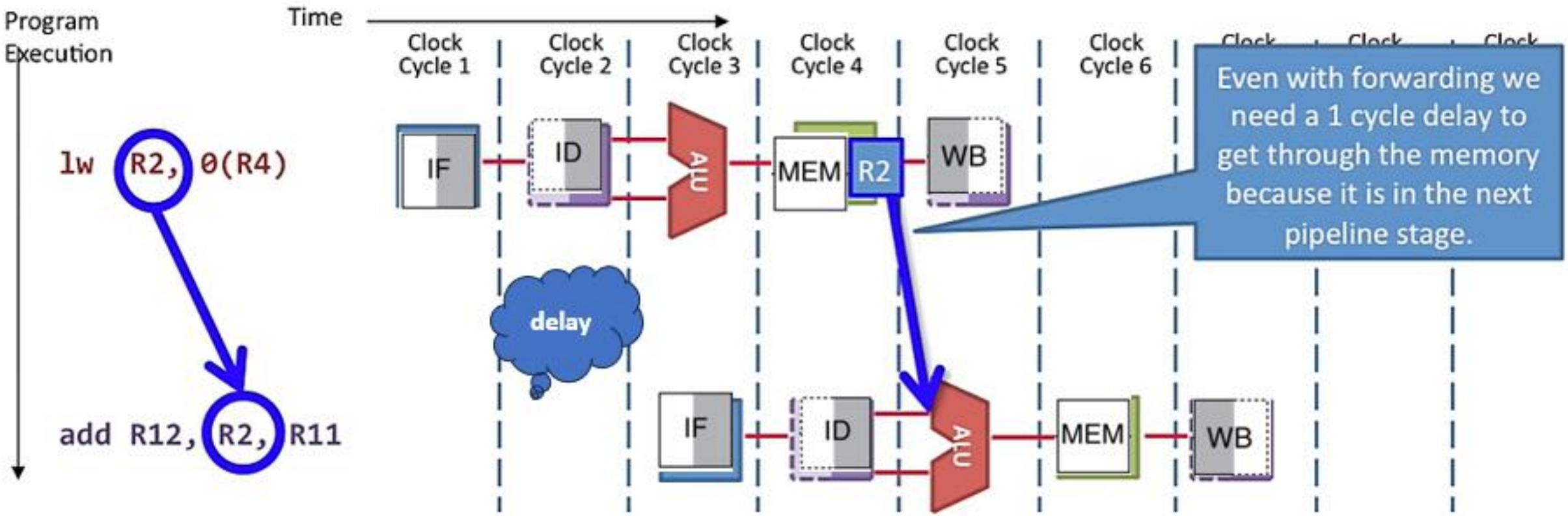
Code Scheduling to Avoid Stalls // example



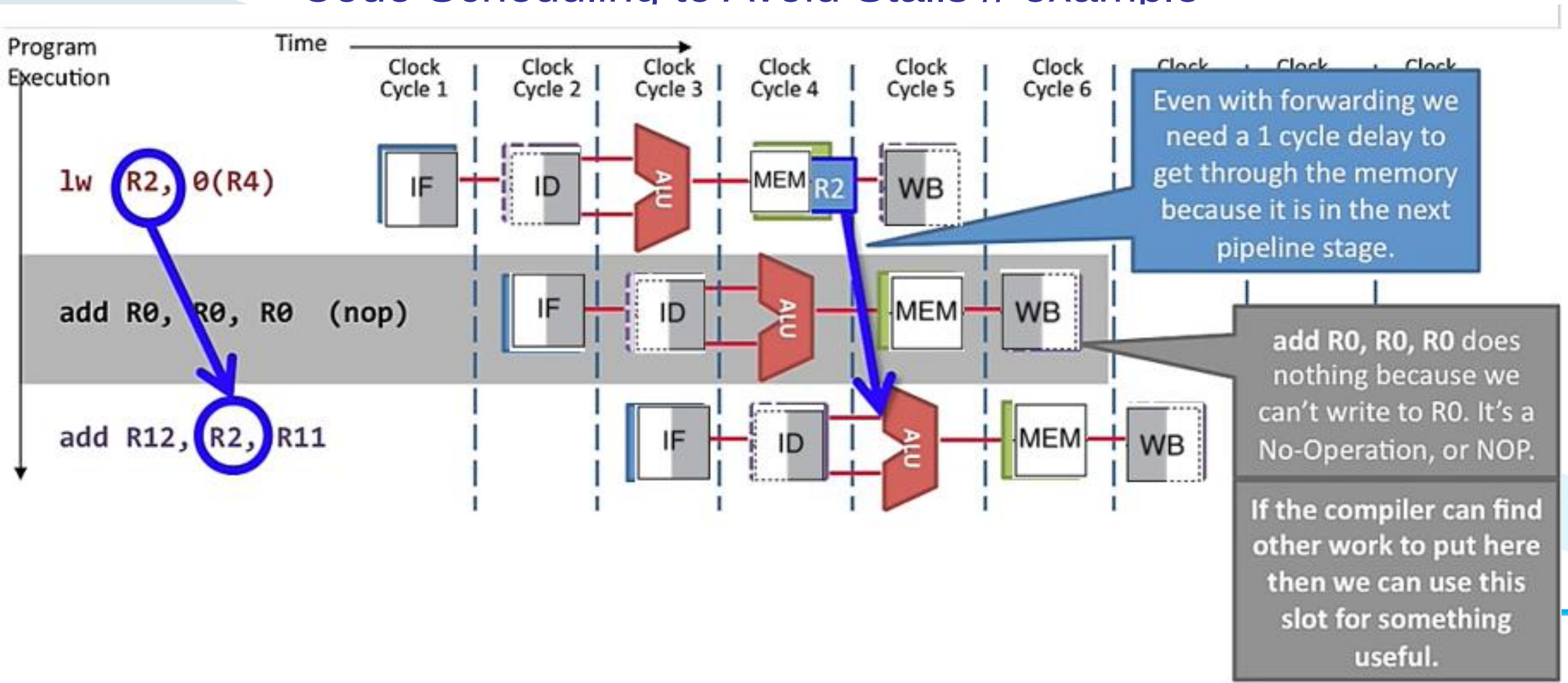
Code Scheduling to Avoid Stalls // example



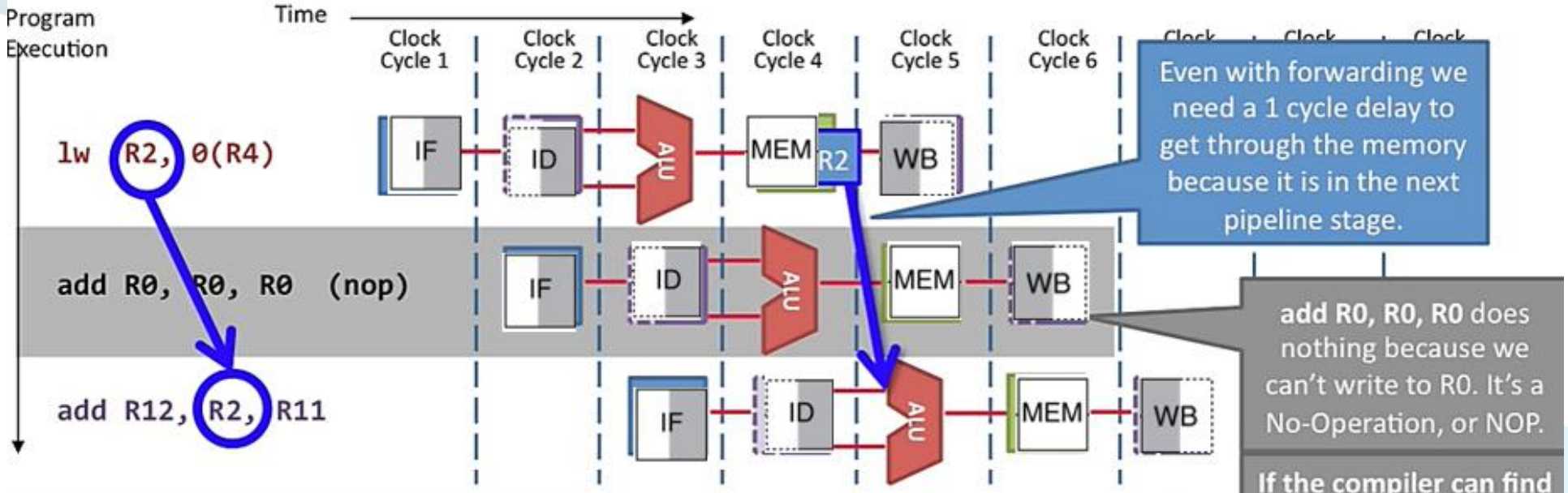
Code Scheduling to Avoid Stalls // example



Code Scheduling to Avoid Stalls // example



Code Scheduling to Avoid Stalls // example



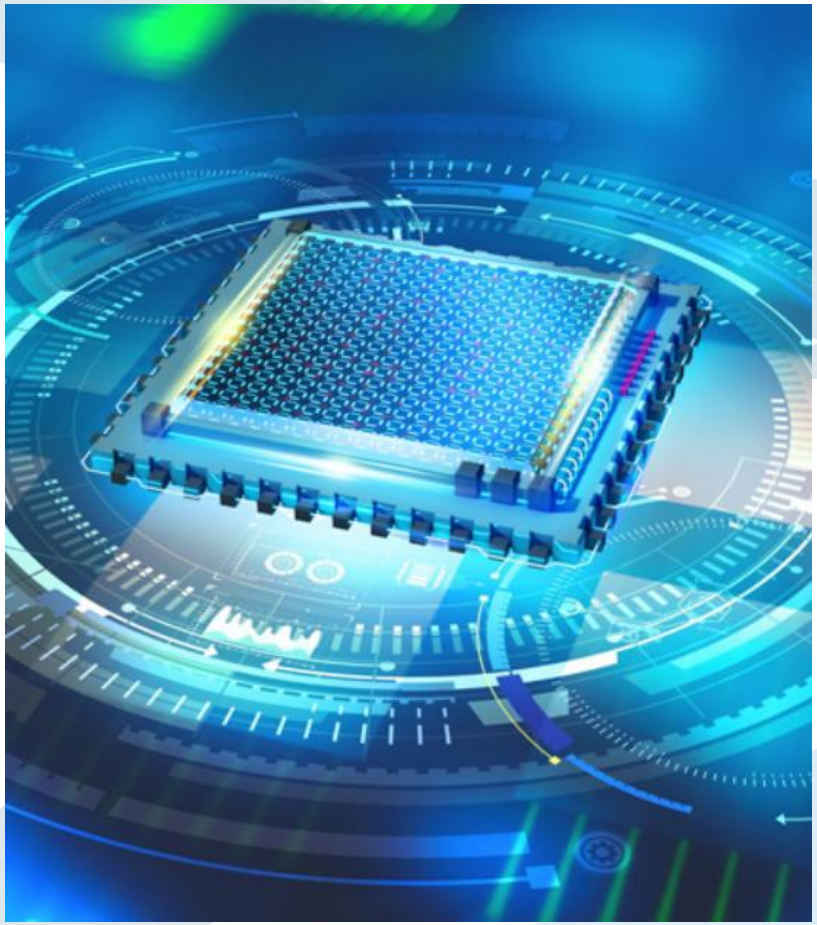
Q: What forwarding is needed from a store instruction?

- Same as for a load instruction
- None
- An extra path in the MEM stage

A: None

Store instructions do not write back to the register file, so no subsequent instructions can be waiting for their results. They may need forwarding **to** the store instruction to get the address or data **from** a previous instruction.

Control Hazards



أخطار التحكم (Control Hazards or Branch hazards):

عندما التعليمة لا تنفذ في دورة الساعة الصحيحة لأن التعليمة التي جلبت غير التي نحتاجها بالتالي تدفق عناوين التعليمات سيكون غير التي يتوقعها الـ pipeline ويظهر من الحاجة لأنشاء قرار مستند على نتيجة تعليمة بينما تنفذ التعليمات الأخرى وبالتالي المخطط الأنبوبي pipeline لا يمكنه معرفة احتمال التعليمة التالية التي ستنفذ حتى يستقبل تعليمة القفز من الذاكرة.

حلول أخطار التحكم:

1. التوقف عند القفز: لنفترض أننا وضعنا تجهيزات إضافية والتي تختبر المسجلات، تحسب قيمة عنوان القفز، تحدث قيمة الـ PC خلال المرحلة الثانية من pipeline.

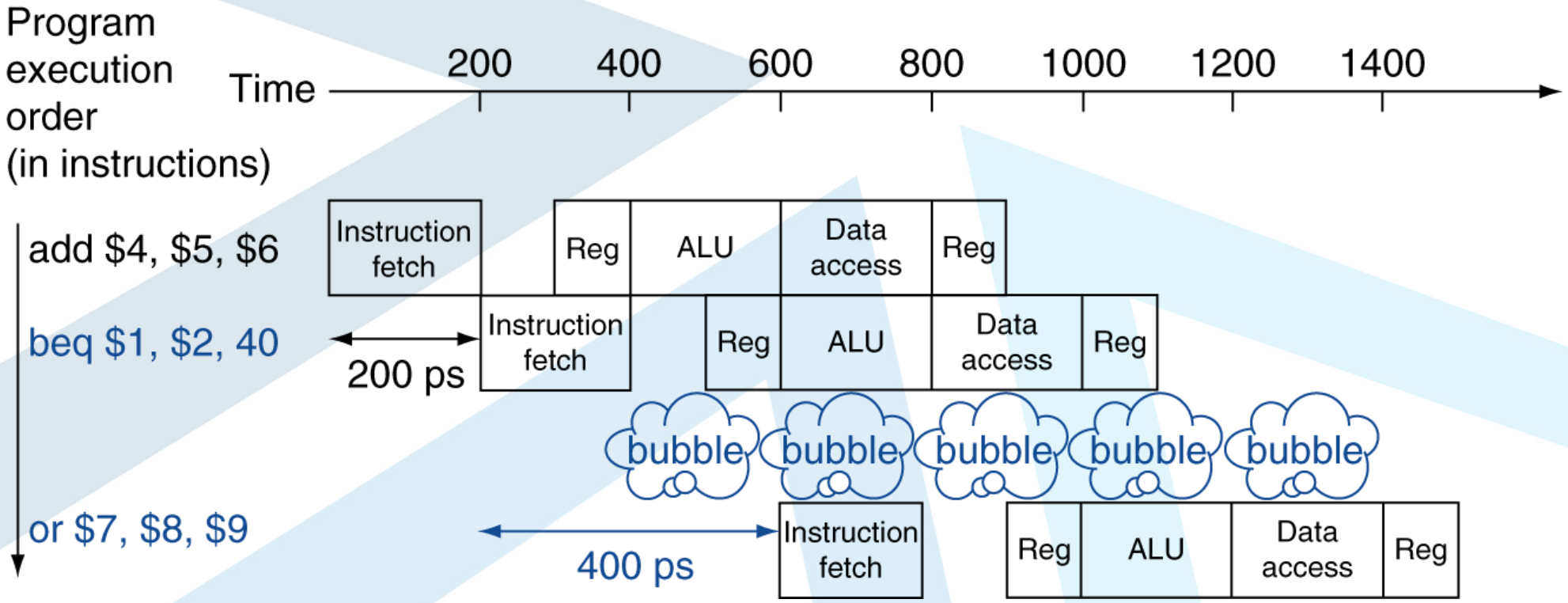
تكلفة هذا الخيار كبيرة جداً وخاصةً:

- عندما لا نستطيع حل القفز في المرحلة التالية (كما هي الحالة في المخططات الطويلة longer pipelines).
- عندما نملك مخطط طويل قرار القفز وحساب القفزة الهدف تؤدى في مراحل متأخرة.

Control Hazards

Stall on Branch

- Wait until branch outcome determined before fetching next instruction



Control Hazards

Stall on Branch

Branch Prediction الحل :

توقع القفز هي طريقة لحل خطر القفز بأن نفترض أننا أعطينا نتيجة القفز ونكمل من هذا الافتراض أفضل من الانتظار للتحقق من أنها النتيجة الفعلية.

توقع القفز الساكن (Static branch):

هناك ثلاث احتمالات : القفز يحدث أو لا يحدث أو التوقع يحدث/لا يحدث باعتبار تعليمة القفز (الدور الاتجاه).

القفزات في أسفل الحلقة: يتم القفز عودة إلى بداية الحلقة يشبه بذلك أن القفز قد أخذ وحصل القفز عودة (إلى الوراء).

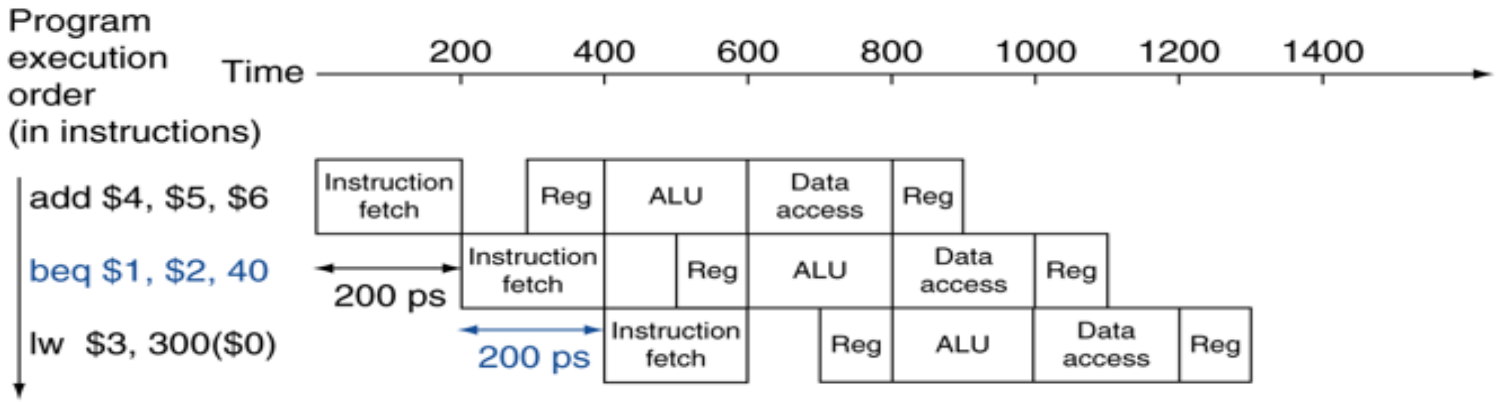
تنبؤ مناسب نتوقع الفروع التي ستقفز إلى عناوين سابقة.

Control Hazards

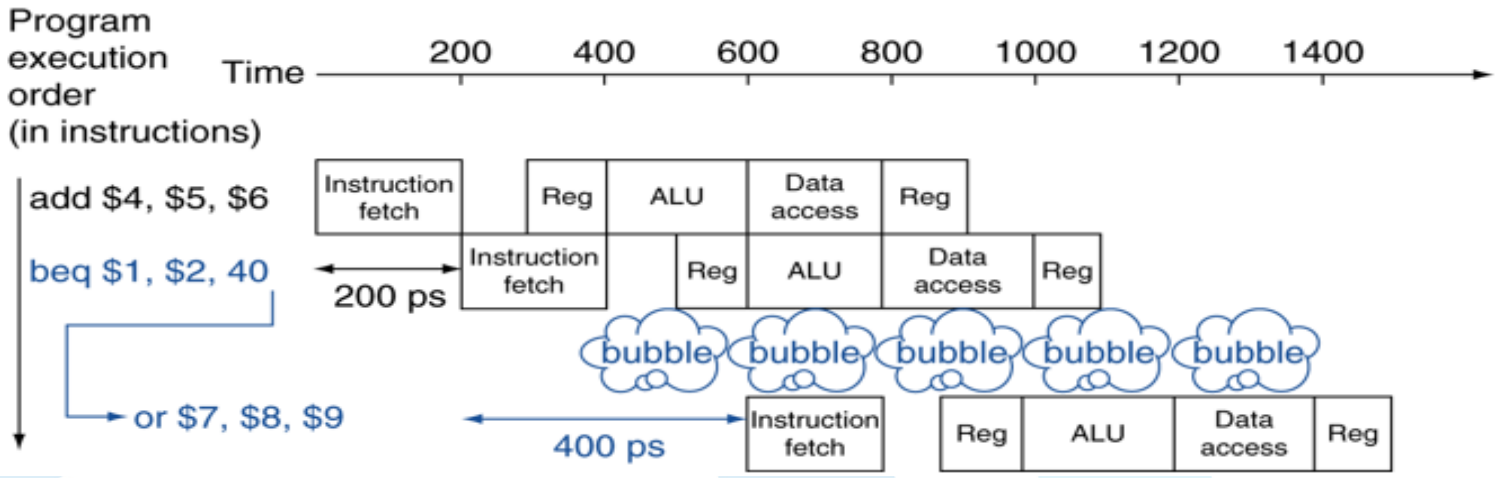
Branch Prediction

MIPS with Predict Not Taken

Prediction correct



Prediction incorrect



Control Hazards

Branch Prediction

More-Realistic Branch Prediction

- Static branch prediction
  - Based on typical branch behavior
  - Example: loop and if-statement branches
    - Predict backward branches taken
    - Predict forward branches not taken
- Dynamic branch prediction
  - Hardware measures actual branch behavior
    - e.g., record recent history of each branch
  - Assume future behavior will continue the trend
    - When wrong, stall while re-fetching, and update history

Example:1

The 5 stages of the processor have the following latencies:

	Fetch	Decode	Execute	Memory	Writeback
a.	300ps	400ps	350ps	550ps	100ps
b.	200ps	150ps	100ps	190ps	140ps

Assume that when pipelining, each pipeline stage costs 20ps extra for the registers between pipeline stages.

I. Non-pipelined processor: what is the cycle time? What is the latency of an instruction? What is the throughput?

II. Pipelined processor: What is the cycle time? What is the latency of an instruction? What is the throughput?

الحل

I. Nonpipelining

a. Cycle time CT=300+400+350+550+100=1700ps

Latency =CT=1700 ps

Throughput= 1/CT= 1/1700 inst/ps

b) CT= 200+150+100+190+140= 780 ps

latency=CT=780 ps

Throughput=1/780 inst/ps.

ملاحظة ١: زمن الدور الأصغري هوCT

ملاحظة ٢ : تردد العمل الأعظمً هو مقلوب الدور الأصغري

F=1/T=1/1700=.....Ghz.

Example:1

	Fetch	Decode	Execute	Memory	Writeback
a.	300ps	400ps	350ps	550ps	100ps
b.	200ps	150ps	100ps	190ps	140ps

II. pipelining processor

a)CT= max(300,400,350,550,100)+20 ps

CT=550+20=570 ps

latency=CTxno. of stages=570x5=2850 ps

Throughput=1/570 inst/ps

b) CT=200+20=220ps

Latency=220x5=1100 ps

Throughput= 1/220 inst/ps

جَامعة
المَنارة
MANARA UNIVERSITY

نهاية المحاضرة التاسعة