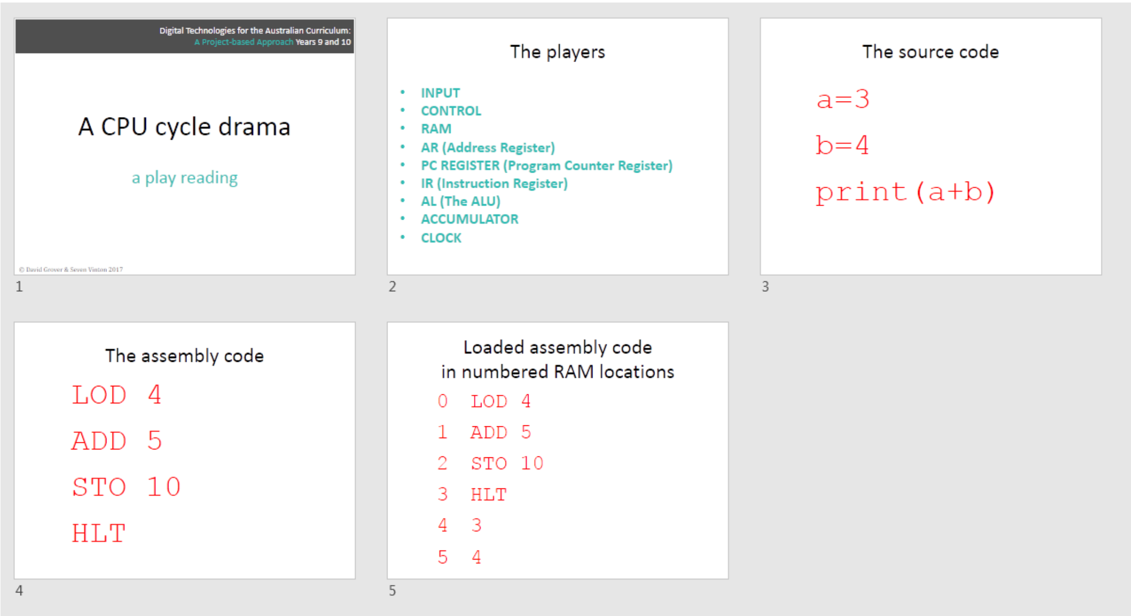
CPU cycle play reading

Your teacher will select eight students to play roles. Each player sits on a seat at the front of the room with their name attached above their head on the wall behind them. Each player first highlights their parts on their scripts.

There is a simple PowerPoint file to be shown during the reading of the play.

Class activity: Play reading

At the end of the play the final slides form a quiz. The class explains the purpose of each the components represented, either as class discussion or as traditional written quiz.



The slide showing the actors is displayed

The players:

- INPUT
- CONTROL
- RAM
- AR (Address Register)
- PC REGISTER (Program Counter Register)
- IR (Instruction Register)
- AL (The ALU)
- ACCUMULATOR
- CLOCK

	[*the slide showing the program to be executed is displayed*]
INPUT:	[*looking towards the slide*] I think something's coming my way. It looks like another program.
CONTROL:	Will we be busy with this one?
INPUT:	[*witheringly*] No, it's just another of those three liners for that beginner computing course. They seem to think it's real programming! a=3 b=4 print(a+b) … Should we treat it seriously?
CONTROL:	Of course. We're always serious anyway. I'll send it to the assembler to translate it into assembly language … so we can follow it more easily! LOD 4 ADD 5 STO 10 HLT 3 4 There! Done! [*the slide showing the assembly code to be executed is displayed*] I'll send the whole lot to RAM.

RAM: OK. I'll store it and I'll give each line an address so we can keep track.

0 LOD 4
1 ADD 5
2 STO 10
3 HLT
4 3
5 4

I've noticed there are integer 3 and 4 at memory locations 4 and 5.

[*waits a beat*]

Done!

[*the slide showing the program loaded in RAM is displayed*]

CONTROL: It looks like we're ready to start.

Registers! Get ready ...

[CONTROL *points at the other characters one by one. They acknowledge* CONTROL *by a nod or thumbs up, except* PC REGISTER *who gives a sarcastic salute and* IR *who does a fist pump into the air.*]

Address register – you'll be minding whatever address is being handled in the current instruction.

Program counter – you'll keep track of the address of the next line of the program.

Instruction register – your job is important. You'll only have to mind one line from RAM but it will be the instruction we're doing at that moment.

Accumulator – you can relax for a while. You'll be storing all the answers ALU gets for us. If you need to add two numbers, you can use register X and register Y to hold them while you add.

Remember, you're a register too, but you won't be busy for at least a few millionths of a second. You're special. Relax!

Ram – send me the first instruction from the first location, 0.

RAM: [*brightly*]

LOD 4

Done!

CONTROL: Do you see it, IR?

IR: I see LOD and I've copied it. [*hesitating*] But I have no idea what it means.

CONTROL: It means to Load but we'll need a location. What do you see AR? Is there a location as well?

AR: Yes! Address location 4.

CONTROL: Program Counter – I hope you're keeping track of where we're up to!

PC REGISTER: I'm pointing to the address of the next instruction: location 1.

CONTROL: Good! Address register, tell us what is at your location 4.

AR: It's an integer; 3 actually.

CONTROL: OK. I'll send that data to the Accumulator.

ACUMULATOR: [*loud/excitable*] Thanks guys! At last something for me! I'll store it.

CONTROL: OK PC, what's next?

PC REGISTER I'm now pointing to address 1.

CONTROL: IR – what's there?

IR: [*matter-of-fact*] I see ADD and I've copied it.

[*then, sheepishly*] But I don't understand it.

CONTROL: It means add a value to whatever is in the accumulator, but we'll need a value. What do you see AR?

AR: I see location 5.

CONTROL: Still keeping track, PC?

PC REGISTER: I'm pointing at the address of the *next* instruction: location 2.

CONTROL: Address register, tell us what value is at that location 5 you're minding.

AR: It's another integer, 4.

CONTROL: OK. I'll add to whatever is already in the Accumulator, which I think was a 3.

I'll ask AL to do that. Go AL!

AL: I'll need to talk to registers X and Y to help me. I only add things to what I've got already.

Done! The result is 7.

CONTROL: AL, send that to the Accumulator.

ACCUMULATOR: Thanks guys! More for me. I'll hold on to that data 7.

CONTROL: What are you seeing PC?

PC REGISTER: OK. I'm now pointing to address 2.

CONTROL: What's there?

IR: I see STO at that address and I've copied it. Don't have any idea what that means though.

CONTROL: It means we'll have to store the answer we got, but we'll need a location. What are you seeing AR?

AR: I see a location 10.

CONTROL: OK, I'll do that Store instruction. Accumulator – send your stored value to location 10.

What are you seeing PC?

PC REGISTER: I'm still pointing to the address of the next instruction: location 3.

[*tongue-in-cheek, winks*] Do you think if I do extra training I might learn to multitask?

CONTROL: What do you see there AR?

AR: I see HLT.

PC REGISTER: [*self-importantly, looking about and then fake-examining nails*]

Is everyone finished? In case you've forgotten I'm still pointing to location 4.

AL: Not interested.

IR: Not interested.

AL: Not interested.

IR: [*covering mouth to disguise a smile*] Not interested.

AL: Not–

CONTROL: [*waves arms to cut them off. Stern expression*] That's all folks.

The answer was 7 and it's sitting there in location 10 as output. Ready for the human when they get round to looking for it.

I hope whoever's out there realises all the steps involved. It's not as simple as it looks!

Clock – how many ticks?

CLOCK: Actually, it was 26 ticks of the clock.

We're a 2.6 Ghz computer, so at 2 600 000 000 cycles every second it took 0.00000001 second.

CONTROL: … a bit slow guys!

[*the rest of the group visibly deflate*]

I know – it's hard to take these beginner programs seriously.

[*shakes head*]

Incidentally, have you seen how thin those new laptops are now?

I *really* need to work out more!