Where do the values that are fed into an operand come from when an instruction is executed using the three address style?	Why are registers faster than memory?
What is the one-address style of instruction? $3$	Describe the load-store style of instruction.
How is code written in ARM Assembly run? $$	What is the instruction to load a value at a memory address into a register?
What is the instruction to store a value in a register to a memory address?	What is the instruction to sum two numbers?

- Implemented using a flip flop or some other very fast volatile storage (rather than smaller, cheaper SRAM).
- Situated inside the processor, so there's less distance for the data to travel, which takes less time.
- Fewer of them so address decoding takes less time
- Data doesn't need to be transferred over a bus.

## Memory.

The resulting value is also copied to a destination address in memory.

The only operations on memory are load and store operations. This means each instruction is very fast and very simple, but there are many instructions.

Where only one memory address may be used in any one instruction. The other operands must be registers.

4

LDR <register> <memory\_address\_alias>

It is first assembled using an assembler into machine code. Then it is loaded into memory and executed sequentially.

5

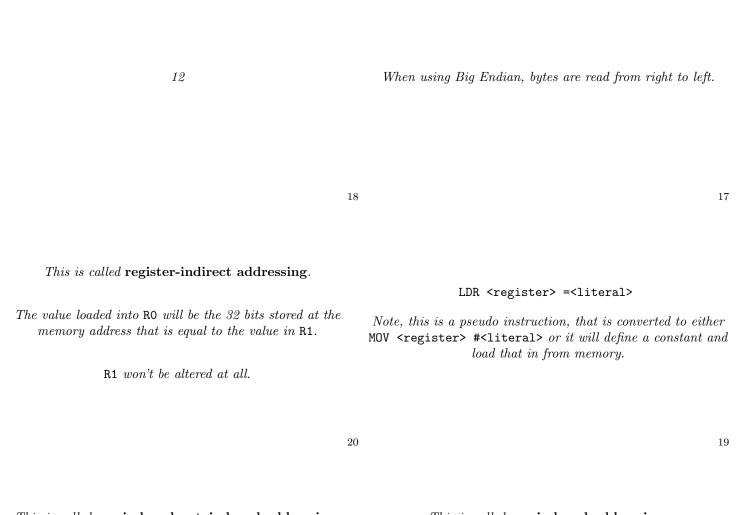
STR <register> <memory\_address\_alias>

1

What is the instruction to branch upon a condition?	What does the program counter do? What register is it?
When is the DEFW command executed. What does it do? What's its syntax?	$What\ does\  exttt{DEFB}\ do?$
What's the syntax of DEFS?	What does STRB do? What is it's syntax?
What does LDRB do? What is it's syntax?	When using Little Endian, bytes are read from .

It is used to store the memory address of the next instruction to be executed. B <condition> <branch\_name> It's register 15. 10 9 It is executed before the program runs. DEFB stores a single byte in memory. If you give it a string, It stores a value at a memory address and assigns the address the whole string will be stored (in multiple bytes). an alias. <alias> DEFW <value\_1>, ... <value\_n> 12 11 Stores the lowest eight bits of a register into memory. <alias> DEFS <number\_of\_bytes>, <value\_of\_bytes> STRB <register> <memory\_alias> 14 13 Loads the lowest eight bits of a specific memory address into a register. The other bits in the register are set to zero. When using Little Endian, bytes are read from left to right. LDRB <register> <memory\_alias>

When using Big Endian, bytes are read from to	How many bits are assigned to a literal in an ARM instruction
17	18
What command should be used to load a literal into a register	In the following instruction, what method of indexing is used, what will RO contain and what will happen to R1?  LDR RO, [R1]
19	20
In the following instruction, what method of indexing is used, what will RO contain and what will happen to R1?  LDR RO, [R1, #4]	In the following instruction, what method of indexing is used, what will RO contain and what will happen to R1?  LDR RO, [R1, #4]!
In the following instruction, what method of indexing is used, what will RO contain and what will happen to R1?  LDR RO, [R1], #4	In the following instruction, what method of indexing is used, what will RO contain and what will happen to R1 and R2?  LDR RO, [R1, R2]



This is called pre-indexed addressing.

This is called pre-indexed addressing.

This is called pre-indexed addressing.

The value loaded into R0 will be the 32 bits stored at the memory address that is equal to the value in R1 + 4.

R1 will be incremented by 4 before the load operation.

This is called pre-indexed addressing.

The value loaded into R0 will be the 32 bits stored at the memory address that is equal to the value in R1 + 4.

R1 won't be altered at all.

22 21

The value loaded into RO will be the 32 bits stored at the memory address that is equal to the value in R1 + R2.

The value loaded into RO will be the 32 bits stored at the memory address that is equal to the value in R1 + 4.

This is called register-indexed addressing.

R1 and R2 will stay the same. R1 will be incremented by 4 after the load operation.

This is called post-indexed autoindexed addressing.

In the following instruction, what method of indexing is used, what will RO contain and what will happen to R1 and R2?

LDR RO, [R1, R2, LSL, #2]

## This is called scaled register-indexed addressing.

The value loaded into RO will be the 32 bits stored at the memory address that is equal to the value in R1 + (R2 \* 4).

R1 and R2 will stay the same.