ARM v6M Load/Store Instructions

- Multiple ARM instruction sets
  - *Focus on v6M Thumb instruction set in this class (16-bit instruction encodings)*
- The ARMv6-M documentation and quick reference guide are provided on Blackboard
- Multiple versions of many instructions exist for different cases
- Three classes of load/store we will use
  - *Load – get from main memory*
  - *Store – save to main memory*
  - *Move – transfer/initialize* ***between CPU registers***

ARM v6M Load Instructions

- Variety of different instruction formats for load
  - *12 different versions for the v6-M Thumb instruction set alone*

- All of them are based off the following two instruction types

LDR Rd, [Rn, #<imm>]

Loads the contents of address Rn + imm into Rd

Note: imm means immediate value (aka a number)

Add # to denote number for the assembler

LDR Rd, [Rn, Rm]

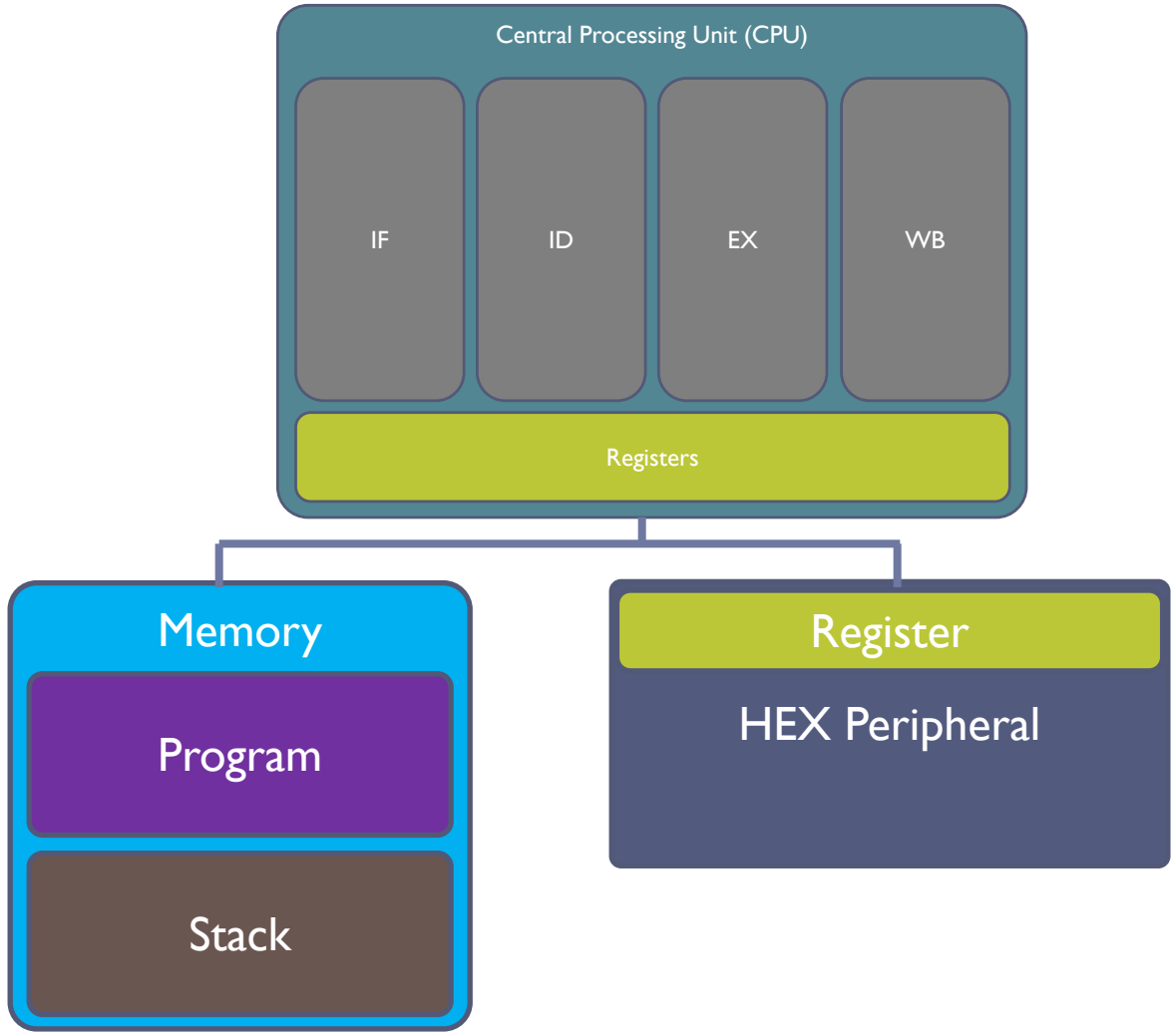
Load the contents of address Rn + Rm into Rd

LDR Rd, [Rn, Rm]

Rd = R0
R1 = 0xFF200000
R2 = 0x50

LDR R0, [R1, R2]

Steps:
1.) CPU adds R2 to R1
0xFF200000 + 0x50 = 0xFF200050
2.) CPU issues request on bus to get contents at address 0xFF200050
3.) Contents of address 0xFF200050 are saved into R0



ARM v6M Load Pseudo Instructions

- Assembler also supports pseudo instructions
  - *These are single line instructions that actual do multiple operations*

- There is a very useful pseudo load instruction

LDR Rd, =<value> (Translates to place value into Rd)

Example: LDR R2, =0xFF200000
0xFF200000 gets stored as a constant value in our program at a given address
Assembler calculates the offset from the address of the constant value to the PC
Translates the instruction to behind the scenes to

LDR R2, [PC, #offset]

ARM v6M Store Instructions

- Variety of different instruction formats for store
  - *8 different versions for the v6-M Thumb instruction set alone*

- All of them are based off the following two instruction types

STR Rd, [Rn, #<imm>]

Store the contents of Rd at address Rn + imm

Note: imm means immediate value (aka a number)

Add # to denote number for the assembler

STR Rd, [Rn, Rm]

Store the contents of Rd at address Rn + Rm

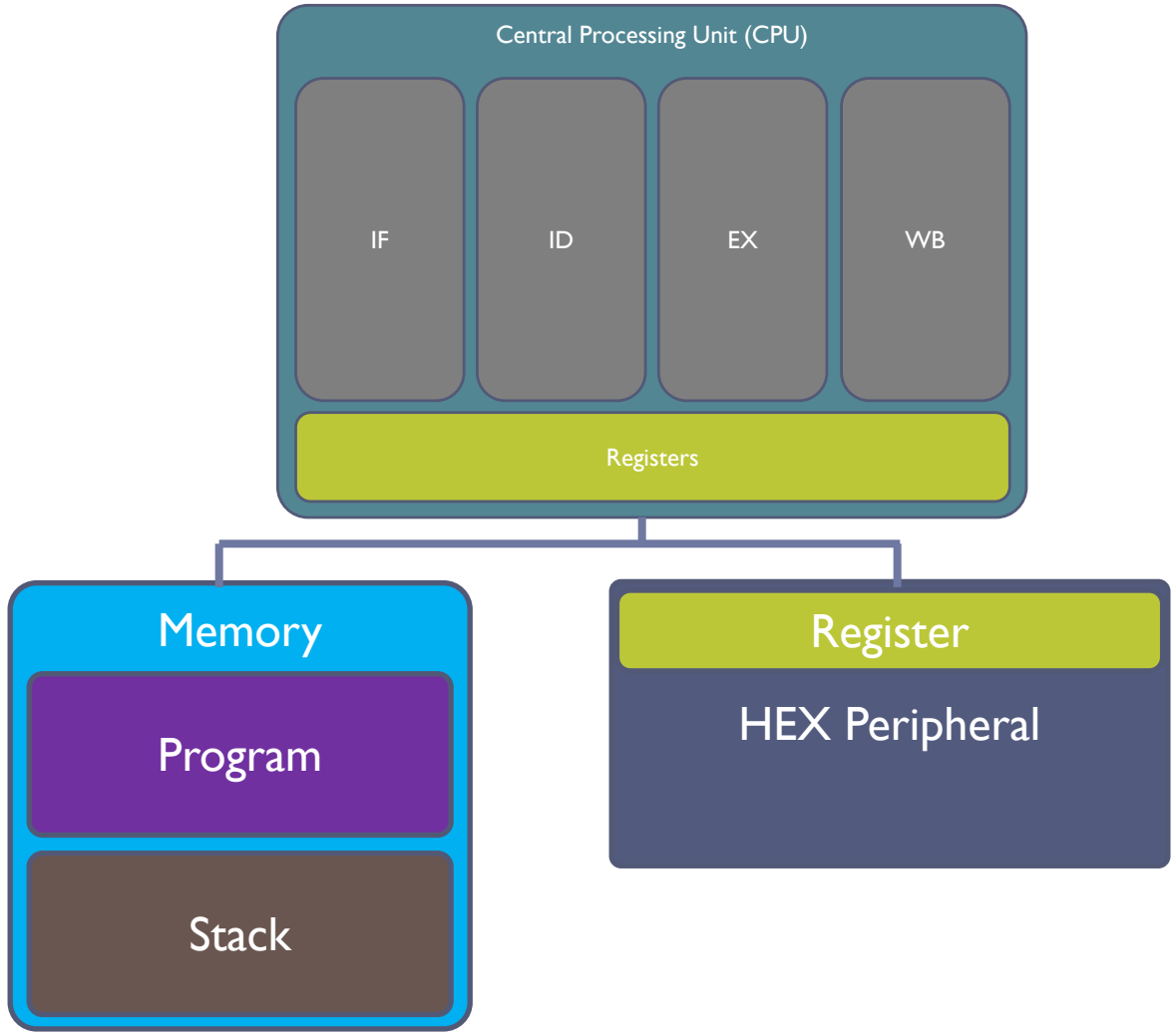
Store Example

R1 = 500
R2 = 50
R3 = 4

STR R2, [R1, R3]

Steps:
1.) CPU adds R1 to R3
500 + 4 = 504

2.) CPU sends value of R1 (500) to address 504. Contents of address 504 now equal 50



ARM v6M Move Instructions

- Variety of different instruction formats for move
  - *5 different versions for the v6-M Thumb instruction set*

- Allows for initializing registers and moving data between registers

MOVS Rd, #<imm>
Move the immediate value into register Rd

MOV Rd, Rm
Move the contents of register Rm into register Rd

MVNS Rd, Rm
Move negative – Invert (Not) the contents of Rm and store at register Rd

Move Example

R2 = 20
R3 = 4 (0x00000004)

MVNS R2, R3

Steps:
1.) CPU Inverts (flips all bits of R3)
0x00000004 -> 0xFFFFFFFB

2.) CPU places the value
0xFFFFFFFB into register R2