ADD X1, X2, X3 ⇒ X1 = X2 + X3

Store Instruction

- The STUR instruction tells the CPU to store (copy) the contents of CPU register to a memory location pointed by destination register.
- Since the CPU registers are 64 – bit (8 byte) wide, we need 8 consecutive memory locations to store the contents of the register.

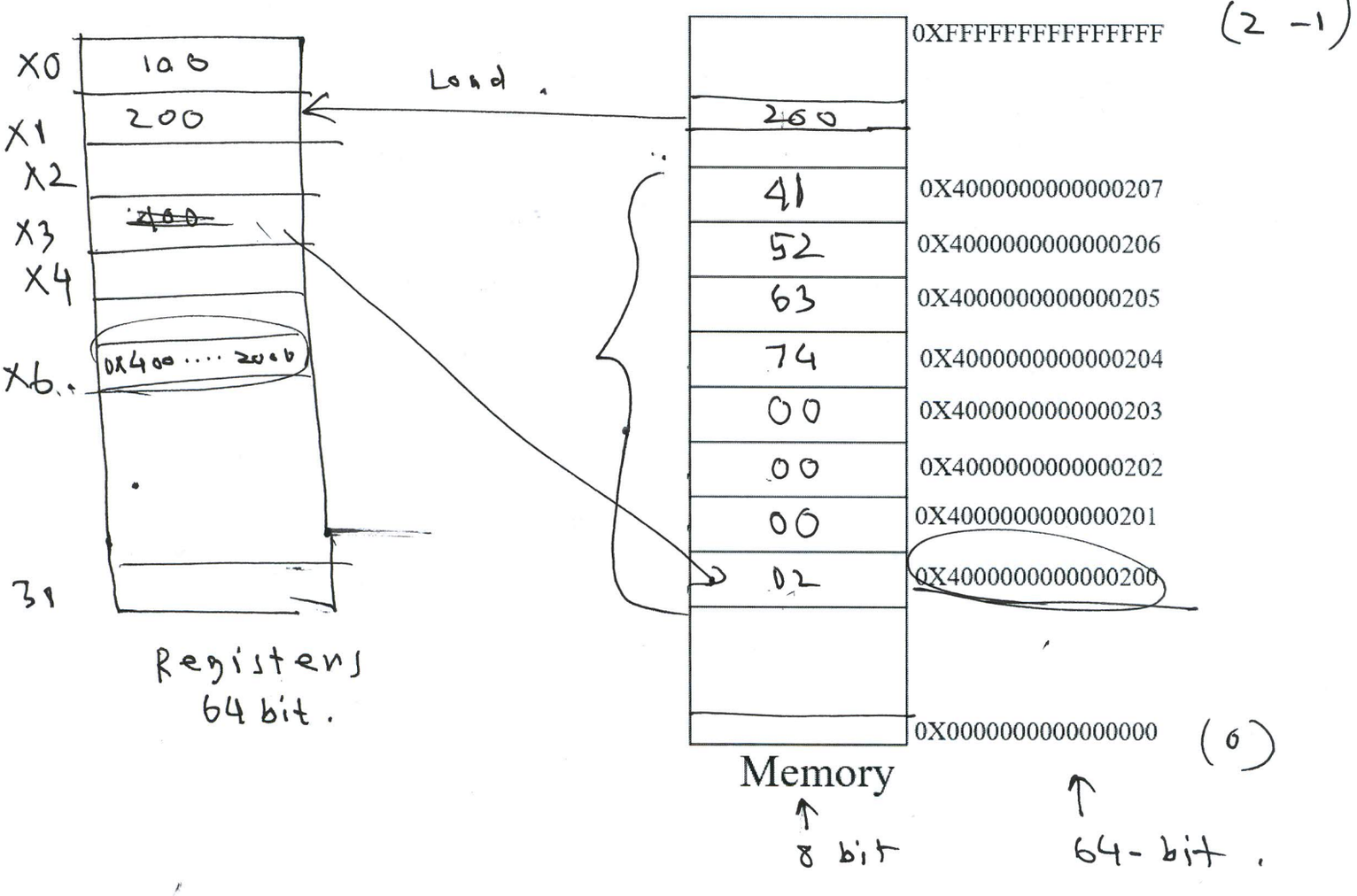
STUR X3, [X6, #0]

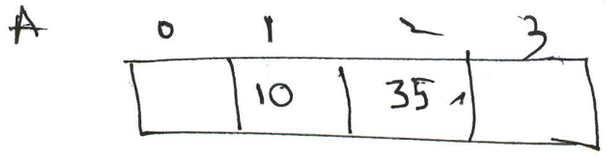
X6 + 0 = X6

1 Hex digit = 4 bit

Hex

Problem: Assume that X6 = 0X4000000000000200 and X3 = 0X4152637400000002. What will happen after running the following instruction: STUR X3, [X6, #0].





assuming
A[1] = 10
h = 25

Example: Convert the following C++ code to LEGv8 Assembly code. Assume A is an Array of 10 doublewords, variable h is associated with register X21, and base address of the array A is in X22.

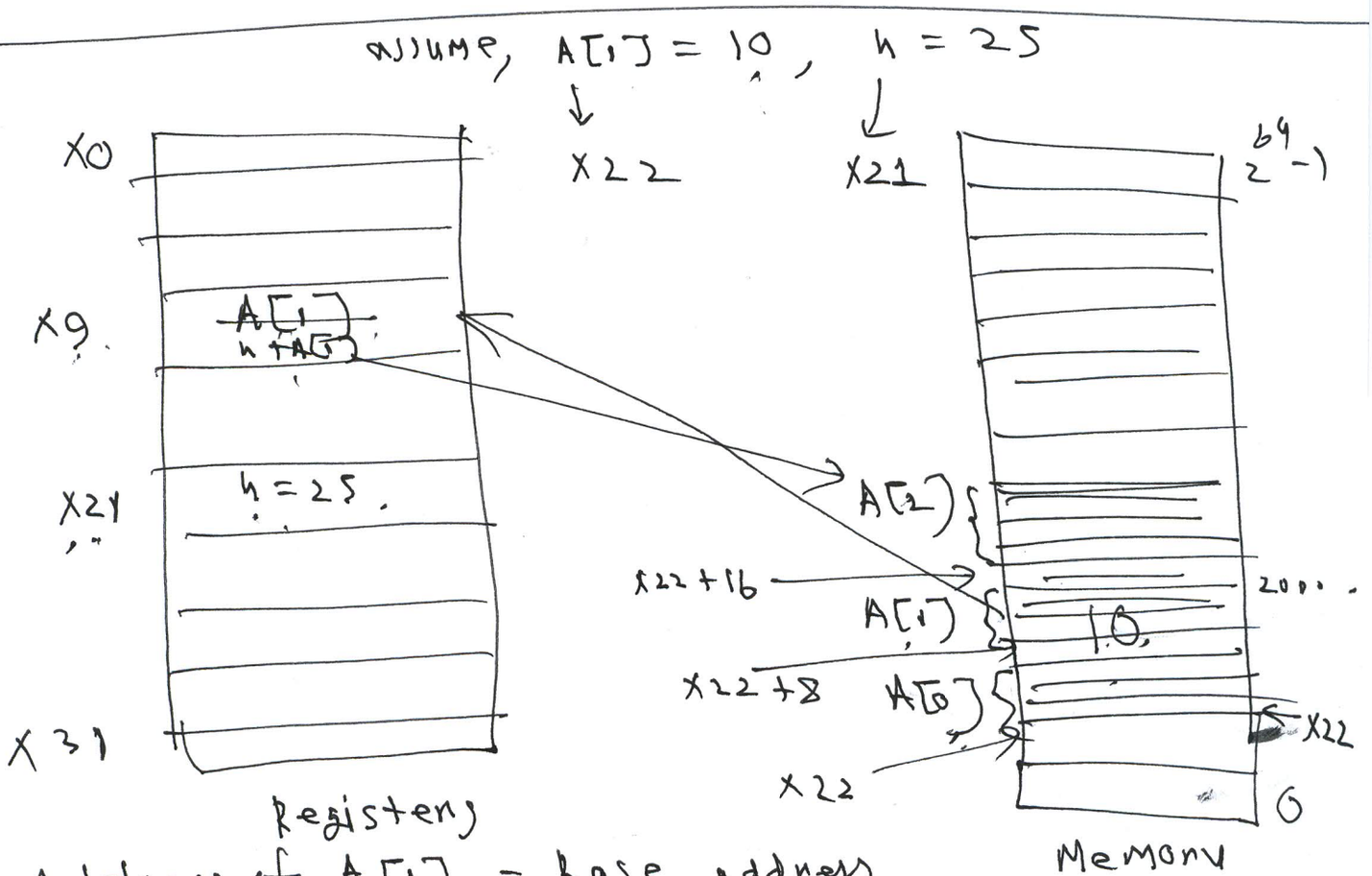
A [2] = h + A [1];

STUR

A [10] → X22

h → X21

```
LDUR   X9, [X22, #8]      // X9 = A[1]
ADD    X9, X9, X21,       // X9 = h + A[1]
STUR   X9, [X22, #16]     // A[2] = h + A[1]
```



Address of A[1] = base address + 1 * 8,
= X22 + 8

a = b

01011

C = a + b

→ 00011

Representing Instructions in the computer (Translating a LEGV8 assembly instruction into a Machine instruction)

Instruction Format: The layout of an instruction is called the instruction format. LEGv8 instructions are 32 bit long.

Instruction	Format	opcode	Rm	shamt	address	op2	Rn	Rd
ADD (add)	R	1112_{ten}	reg	0	n.a.	n.a.	reg	reg
SUB (subtract)	R	1624_{ten}	reg	0	n.a.	n.a.	reg	reg
ADDI (add immediate)	I	580_{ten}	reg	n.a.	constant	n.a.	reg	n.a.
SUBI (sub immediate)	I	836_{ten}	reg	n.a.	constant	n.a.	reg	n.a.
LDUR (load word)	D	1986_{ten}	reg	n.a.	address	0	reg	n.a.
STUR (store word)	D	1984_{ten}	reg	n.a.	address	0	reg	n.a.

LEGV8 R-Format Instructions: (Addition, subtraction. ADD, SUB

opcode	Rm	shamt	Rn	Rd
11 bits	5 bits	6 bits	5 bits	5 bits

32 bits =

ADD X9, X20, X21 X9 = X20+21

opcode = operation code 1112_{10}

Rm = operand 2 21_{10}

shamt = shift amount 0

Rn = operand 1 20_{10}

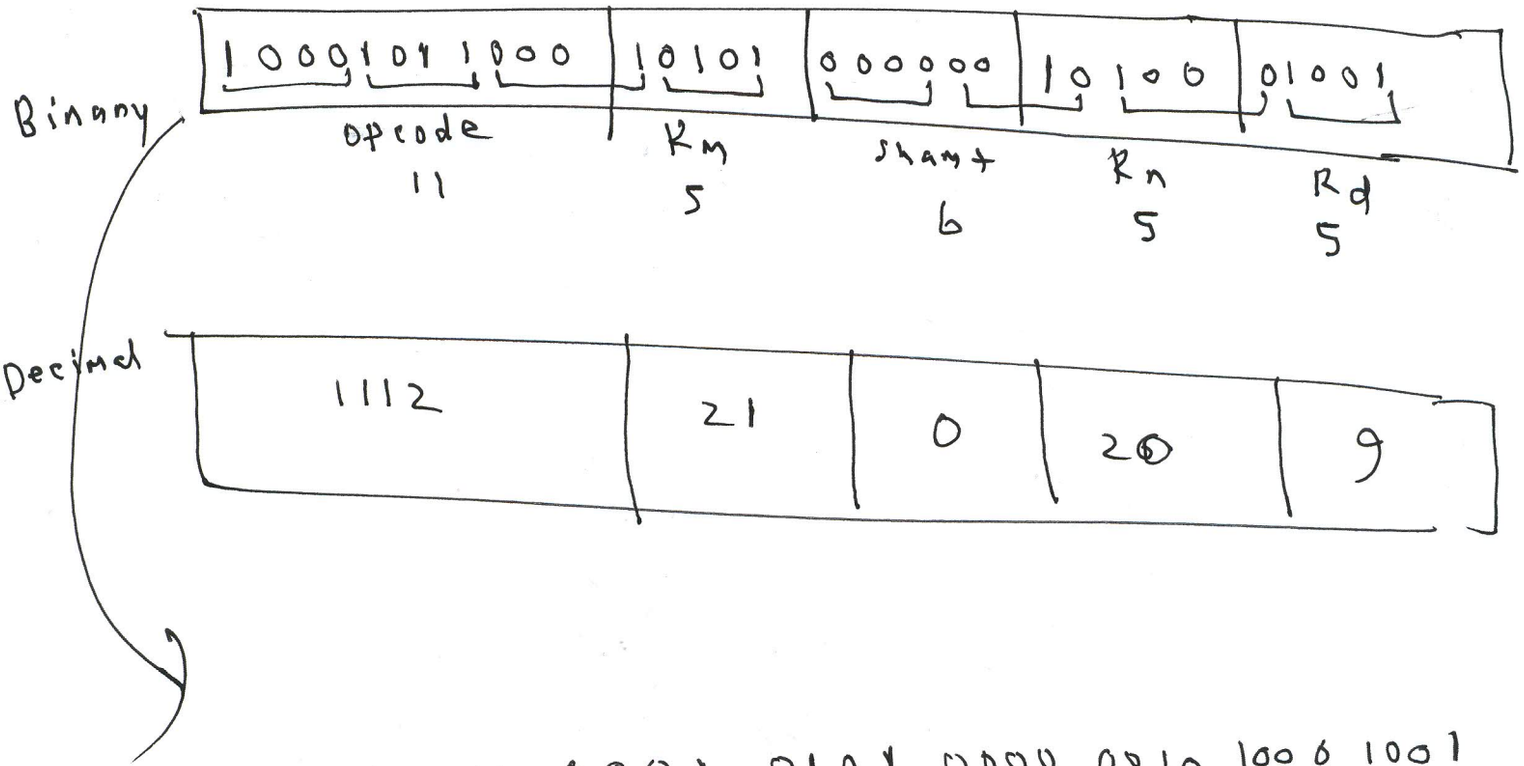
Rd = Destination Register - 9_{10}

$21 = \begin{matrix} 16 & 8 & 4 & 2 & 1 \\ 1 & 0 & 1 & 0 & 1 \end{matrix}$

Translate the following LEGV8 assembly instruction into a machine instruction: **ADD X9, X20, X21**

↓ Rd operand 1 ↓ Rn ← Rm operand2

$1112_{10} = 10001011000$



1000 1011 0001 0101 0000 0010 1001 1001

8 B 1 5 0 2 8 9

HEX = 0X 8B150289

↑ Hex