Instruction set

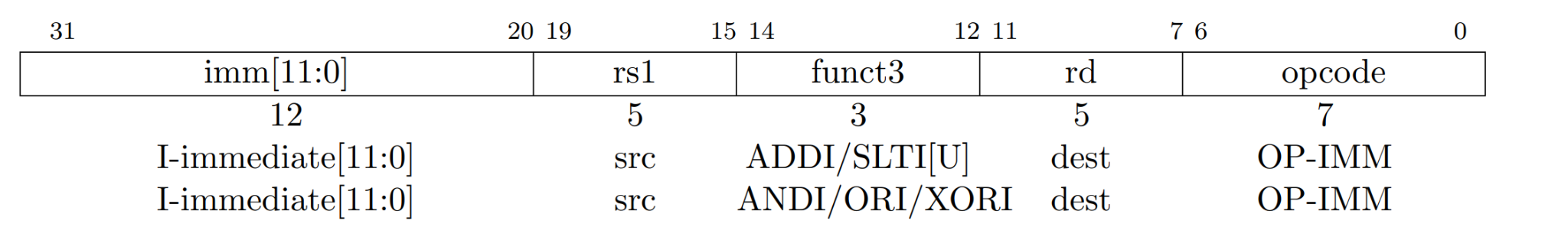
| 31-25 | 24-20 | 19-15 | 14-12 | 11-7 | 6-2 | 1- 0 | instr |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0000000 | rs2 | rs1 | 000 | rd | 01100 | 11 | ADD |
| 0100000 | rs2 | rs1 | 010 | rd | 01100 | 11 | SUB |
| imm[11:0] | | rs1 | 000 | rd | 00100 | 11 | ADDI |
| 0000000 | shamt[4:0] | rs1 | 000 | rd | 10000 | 11 | SLLI |
| imm[11:0] | | rs1 | 000 | rd | 00000 | 11 | LOAD |
| offset[11:5] | rs2 | rs1 | 000 | offset[4:0] | 01000 | 11 | STORE |
| offset[12,10:5] | rs2 | rs1 | 100 | offset[4:1,11] | 11000 | 11 | BLT |
| offset[12,10:5] | rs2 | rs1 | 101 | offset[4:1,11] | 11000 | 11 | BGE |
| offset[11:0] | | rs1 | 000 | rd | 11001 | 11 | JALR |
| offset[20:1] | | | | rd | 11011 | 11 | JAL |
| imm[31:12] | | | | rd | 00101 | 11 | AUIPC |
| imm[31:12] | | | | rd | 01101 | 11 | LUI |
| 0000\_0000\_0000 | | 00000 | 000 | 00000 | 00100 | 11 | NOP |

1. I-type instruction

ADDI

Format: addi rd, rs1, imm

Function: Add rs1 to the immediate value and store the result in rd



| 31-20 | 19-15 | 14-12 | 11-7 | 6-2 | 1- 0 | instr |
| --- | --- | --- | --- | --- | --- | --- |
| imm[11:0] | rs1 | 000 | rd | 00100 | 11 | ADDI |

Li(load immediate: pseudo instruction)

In our assembly code, we have two Li instructions:

Li, a4, 4

Li a5, 3

They are all converted to addi instructions:

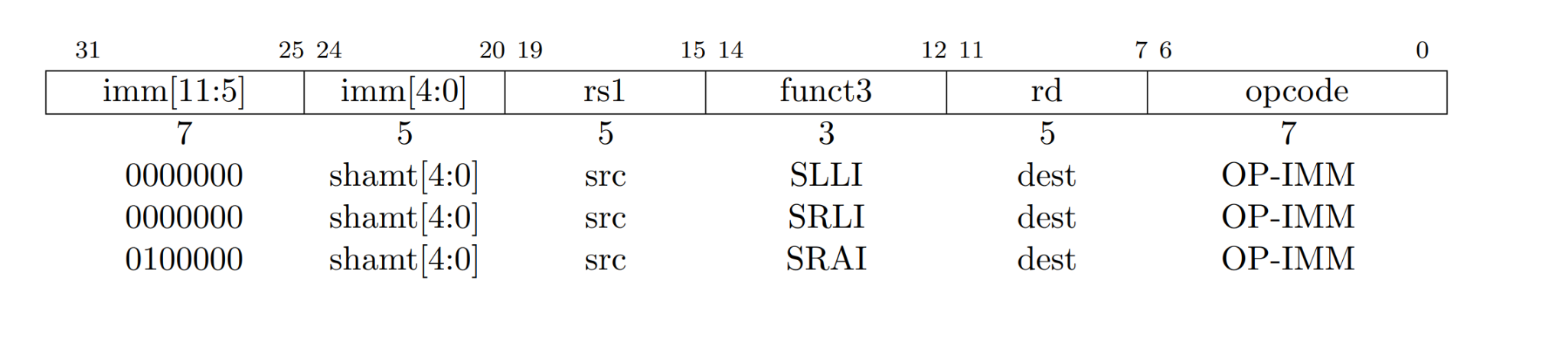
addi a4, x0, 4

addi a5, x0, 3

SLLI

Format: slli rd, rs1, imm

Function: logically shift rs1 left by imm bits and store the result in rd

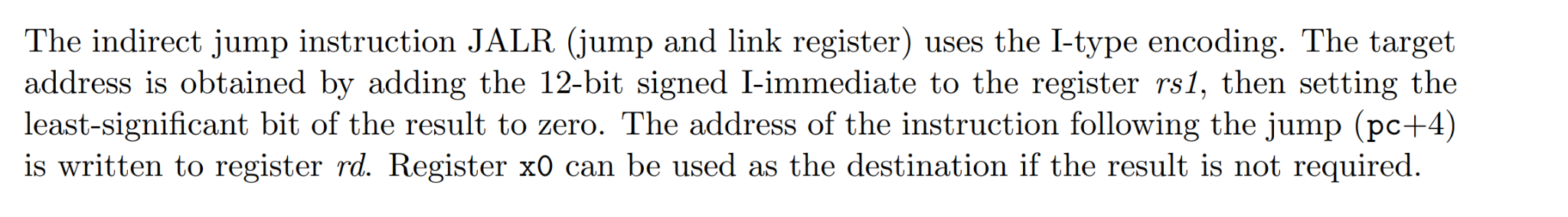


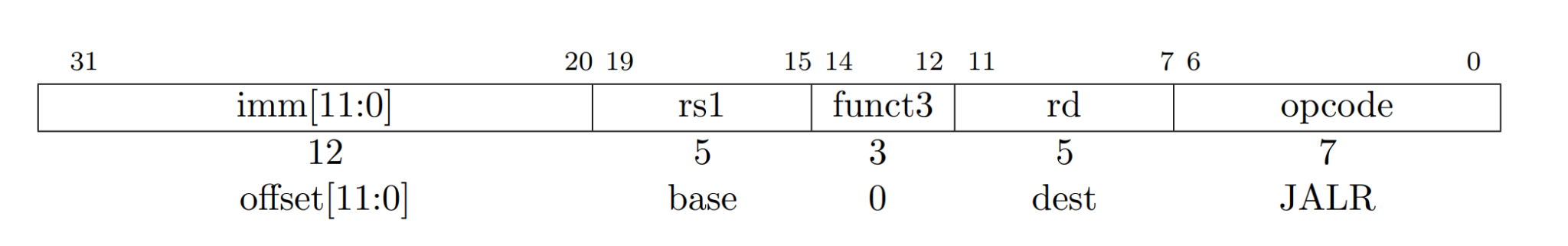
| 31-25 | 24-20 | 19-15 | 14-12 | 11-7 | 6-2 | 1- 0 | instr |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0000000 | shamt[4:0] | rs1 | 000 | rd | 10000 | 11 | SLLI |

JALR

Format: jalr rd, (offset)rs1

Function:





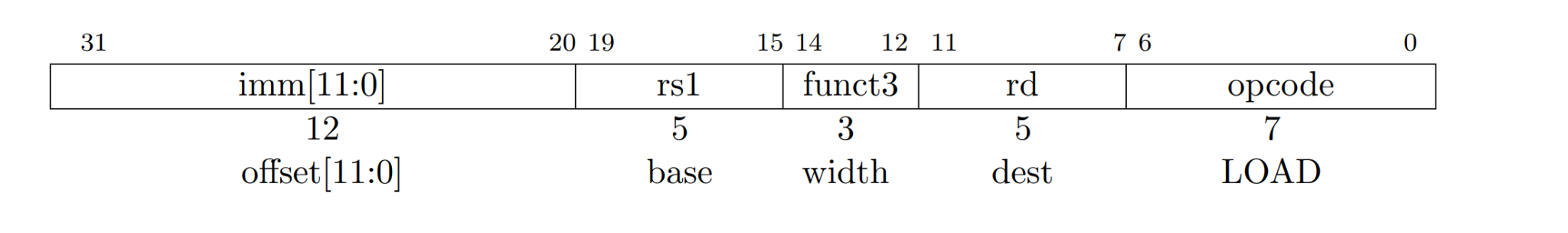
| 31-20 | 19-15 | 14-12 | 11-7 | 6-2 | 1- 0 | instr |
| --- | --- | --- | --- | --- | --- | --- |
| offset[11:0] | rs1 | 000 | rd | 11001 | 11 | JALR |

In our assembly code, we only have Jla, a pseudo instruction, which can be encoded as:

jalr x0, offset(rs1)

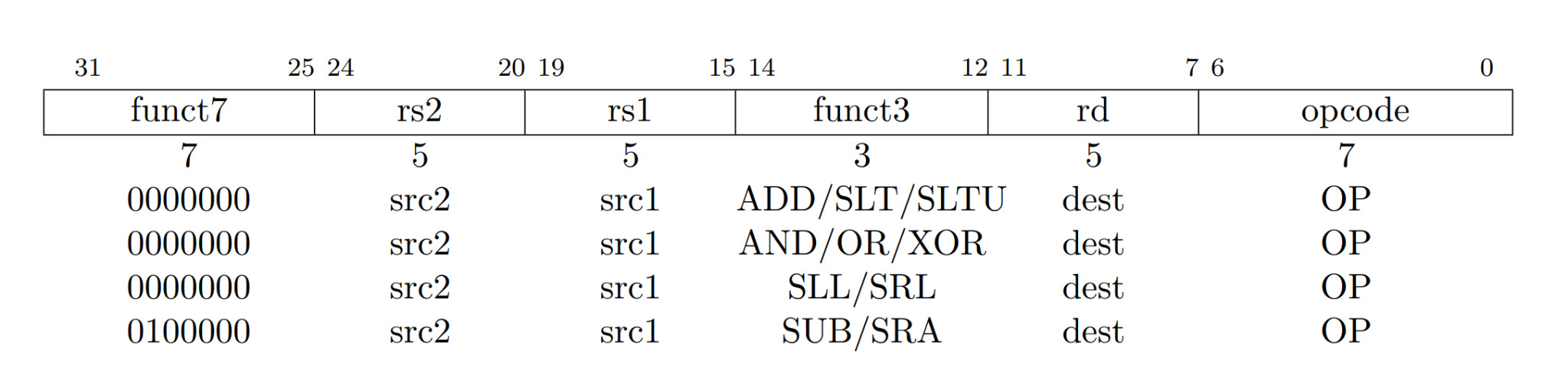
LOAD

Format: load rd, (offset)rs1



| 31-20 | 19-15 | 14-12 | 11-7 | 6-2 | 1- 0 | instr |
| --- | --- | --- | --- | --- | --- | --- |
| offset[11:0] | rs1 | 000 | rd | 00000 | 11 | LOAD |

2. R-type instruction



ADD

Format: add rd, rs1, rs2

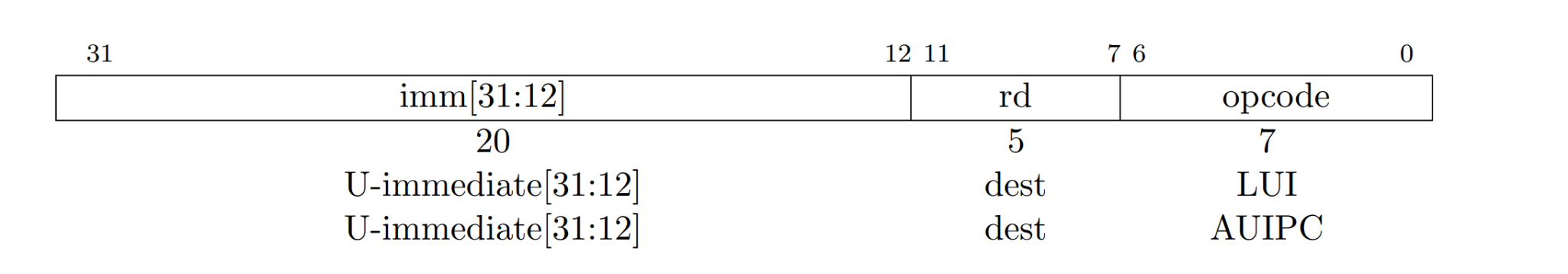
| 31-25 | 24-20 | 19-15 | 14-12 | 11-7 | 6-2 | 1- 0 | instr |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0000000 | rs2 | rs1 | 000 | rd | 01100 | 11 | ADD |

SUB

Format: sub rd, rs1, rs2

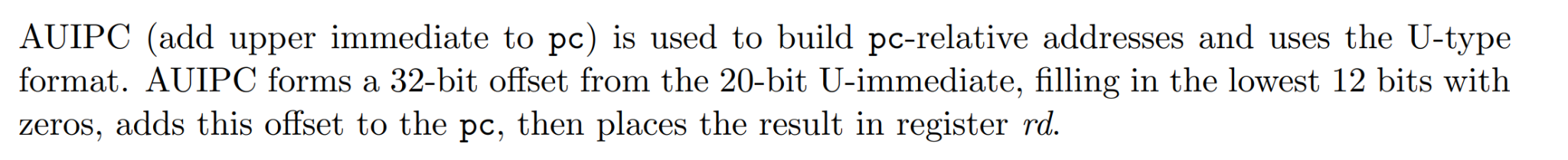
| 31-25 | 24-20 | 19-15 | 14-12 | 11-7 | 6-2 | 1- 0 | instr |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0100000 | rs2 | rs1 | 010 | rd | 01100 | 11 | SUB |

3. U-type instruction



AUIPC

Format: auipc rd, imm



| 31-12 | 11-7 | 6-2 | 1-0 | instr |
| --- | --- | --- | --- | --- |
| imm[31:12] | rd | 00101 | 11 | AUIPC |

In our assembly code, we have the following code:

lla a5, .LC0

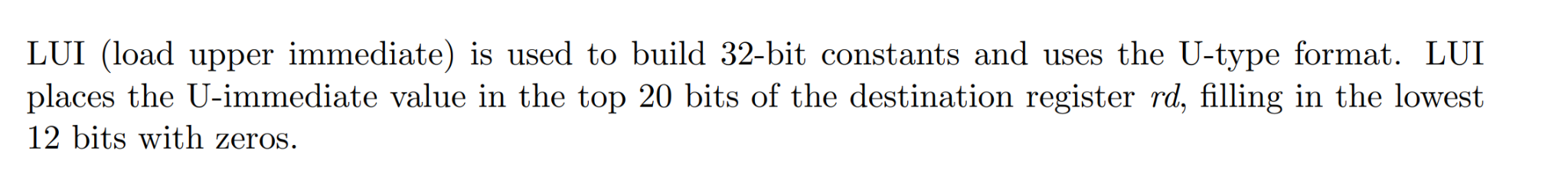
It is converted to:

auipc a5,0x0

mv a5,a5 (addi a5, a5, 0)

LUI

Format: lui rd, imm



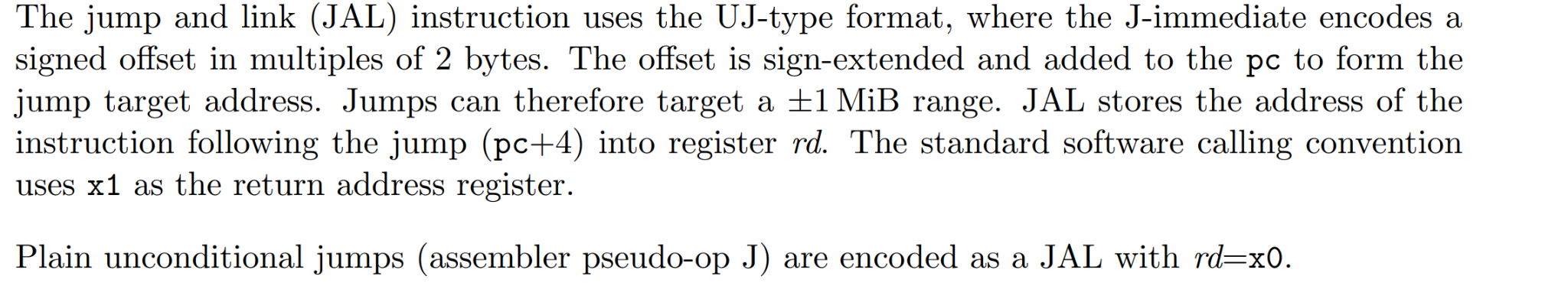
| 31-12 | 11-7 | 6-2 | 1-0 | instr |
| --- | --- | --- | --- | --- |
| imm[31:12] | rd | 01101 | 11 | LUI |

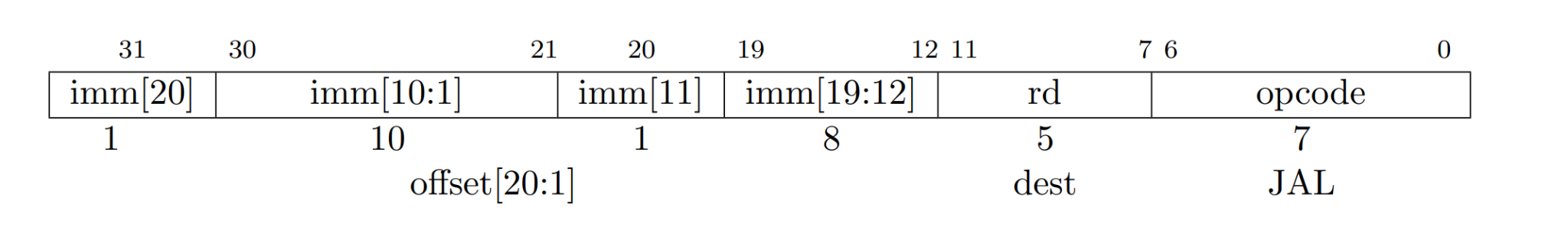
4.UJ-type instruction

JAL

Format: jal rd, offset

Function:





| 31-12 | 11-7 | 6-2 | 1-0 | instr |
| --- | --- | --- | --- | --- |
| offset[20:1] | rd | 11011 | 11 | JAL |

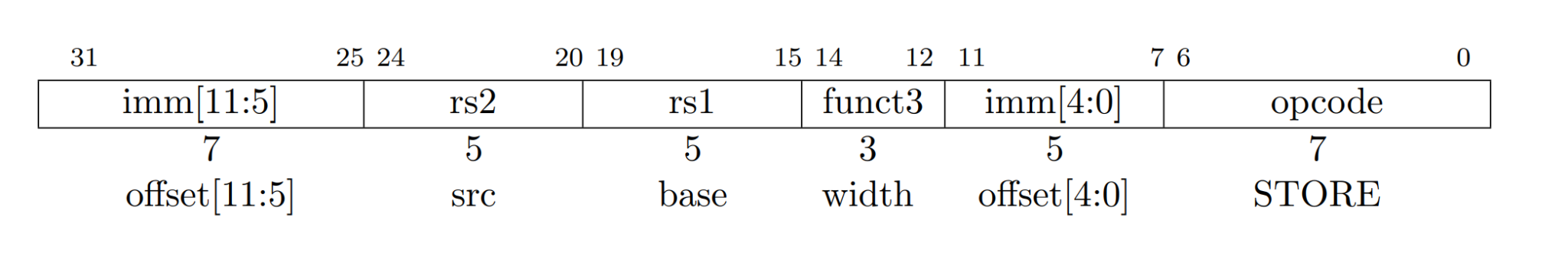
In our assembly code, we only have J, a pseudo instruction, which can be encoded as:

JAL x0, offset

5. S-type instruction

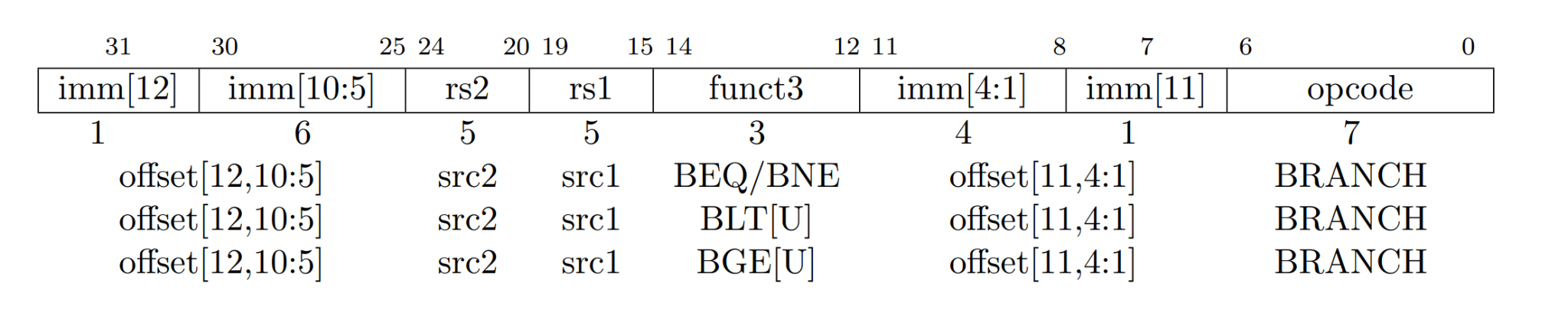
STORE

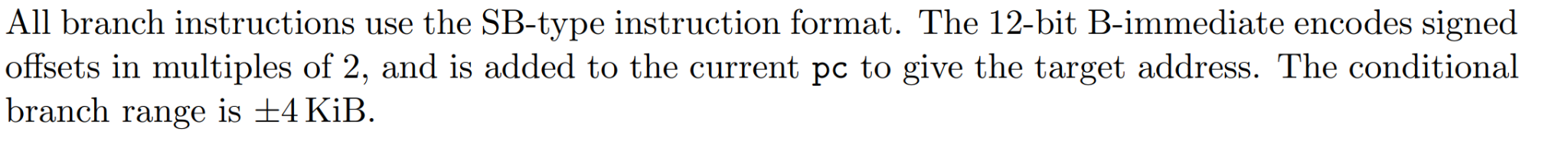
Format: store rs2, (offset)rs1



| 31-25 | 24-20 | 19-15 | 14-12 | 11-7 | 6-2 | 1- 0 | instr |
| --- | --- | --- | --- | --- | --- | --- | --- |
| offset[11:5] | rs2 | rs1 | 000 | offset[4:0] | 01000 | 11 | STORE |

6. SB-type instruction





BLT

Format : blt rs1, rs2, (offset)pc

| 31-25 | 24-20 | 19-15 | 14-12 | 11-7 | 6-2 | 1- 0 | instr |
| --- | --- | --- | --- | --- | --- | --- | --- |
| offset[12,10:5] | rs2 | rs1 | 100 | offset[4:1,11] | 11000 | 11 | BLT |

BGE

Format: bge rs1, rs2, (offset)pc

| 31-25 | 24-20 | 19-15 | 14-12 | 11-7 | 6-2 | 1- 0 | instr |
| --- | --- | --- | --- | --- | --- | --- | --- |
| offset[12,10:5] | rs2 | rs1 | 101 | offset[4:1,11] | 11000 | 11 | BGE |

In our assembly, we have the following codes:

ble a4,a5,.L4

ble a4,a5,.L6

They are converted to:

bge a5,a4,dc <.L4>

bge a5,a4,44 <.L6>

7. NOP instruction

