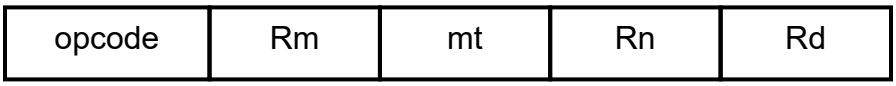
EGC-442 **Introduction to Computer Architecture** *Dr. Izadi*

Course Design Project (100 Points)

I. Overview

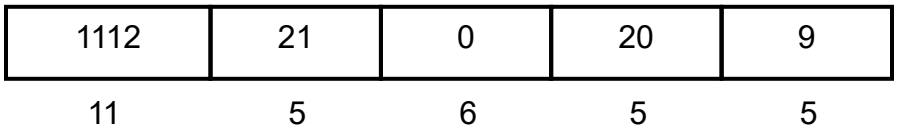
ARM LEGv8 processor is an especial version of ARM processor with the following characteristics:

1. Register file: LEGv8 has a 32 × 64-bit register file
   - 64-bit data is called a "doubleword"
     - 31 x 64-bit general purpose registers X0 to X30
   - 32-bit data called a "word"
     - 31 x 32-bit general purpose sub-registers W0 to W30
   - X0 – X7: procedure arguments/results
   - X8: indirect result location register
   - X9 – X15: temporaries
   - X16 – X17 (IP0 – IP1): may be used by linker as a scratch register, other times as temporary register
   - X18: platform register for platform independent code; otherwise a temporary register
   - X19 – X27: saved
   - X28 (SP): stack pointer
   - X29 (FP): frame pointer
   - X30 (LR): link register (return address)
   - XZR (register 31): the constant value 0
2. LEGv8 Instruction set:
   a. LEGv8 R-format Instructions

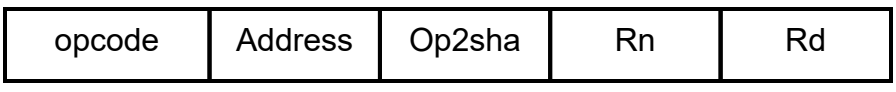


   - opcode: operation code
   - Rm: the second register source operand
   - shamt: shift amount (00000 for now)
   - Rn: the first register source operand
   - Rd: the register destination

   Example: ADD X9,X20,X21

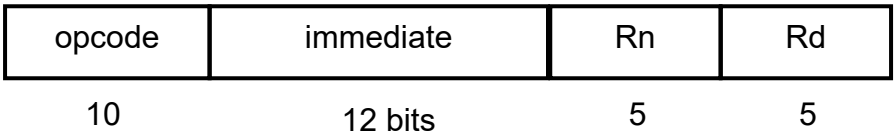


   b. LEGv8 D-format Instructions



- Load/store instructions, LDUR/STUR
  - Rn: base register
  - address: constant offset from contents of base register (+/- 32 doublewords)
  - Rt: destination (load) or source (store) register number

c. LEGv8 I-format Instructions



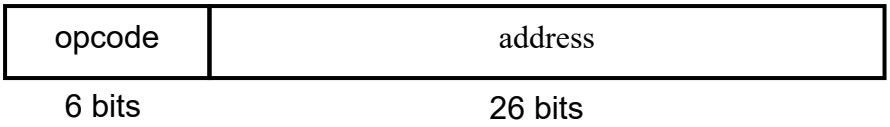
- Immediate instructions
  - Rn: source register
  - Rd: destination register
  - Immediate field is zero-extended

d. LEGv8 Branch instructions

- Conditional branch: Branch to a labeled instruction if a condition is true. Otherwise, continue sequentially

  - CBZ register, L1
    - if (register == 0) branch to instruction labeled L1;
  - CBNZ register, L1
    - if (register != 0) branch to instruction labeled L1;

- Unconditional branch:



- B L1
  - branch unconditionally to instruction labeled L1;
    - Update PC with concatenation of Top 4 bits of old PC
    - 26-bit jump address, plus 00

Example:

```
        SUB     X9,X22,X23
        CBNZ    X9,Else
        ADD     X19,X20,X21
        B       Exit
Else:   SUB     X9,X22,x23
Exit:
```

Table 1: LEGv8 Encoding Summary

Name	Fields							Comments
Field size		6 to 11 bits	5 to 10 bits	5 or 4 bits	2 bits	5 bits	5 bits	All LEGv8 instructions are 32 bits long
R-format	R	opcode	Rm	shamt		Rn	Rd	Arithmetic instruction format
I-format	I	opcode	immediate			Rn	Rd	Immediate format
D-format	D	opcode	address		op2	Rn	Rt	Data transfer format
B-format	B	opcode	address					Unconditional Branch format
CB-format	CB	opcode	address				Rt	Conditional Branch format
IW-format	IW	opcode	immediate				Rd	Wide Immediate format

Table 2: Comparing Instruction format of LEGv8, MIPS, and ARMv7

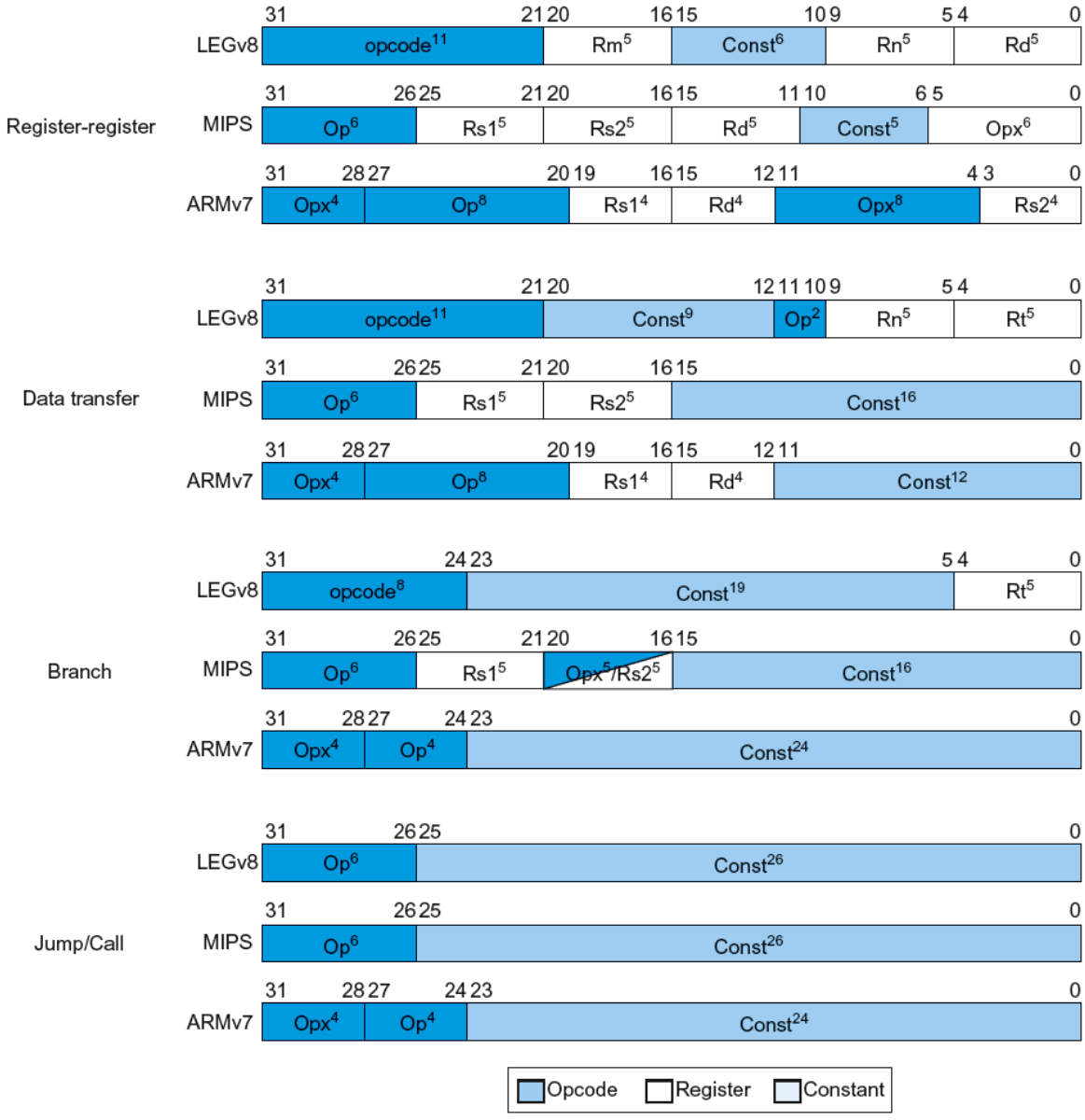


Table 3: Machine Op-code of LEGv8 instructions

Instruction	Opcode	Opcode Size	11-bit opcode range		Instruction Format
			Start	End	
B	000101	6	160	191	B - format
STURB	00111000000	11	448		D - format
LDURB	00111000010	11	450		D - format
B.cond	01010100	8	672	679	CB - format
ORRI	1011001000	10	712	713	I - format
EORI	1101001000	10	840	841	I - format
STURH	01111000000	11	960		D - format
LDURH	01111000010	11	962		D - format
AND	10001010000	11	1104		R - format
ADD	10001011000	11	1112		R - format
ADDI	1001000100	10	1160	1161	I - format
ANDI	1001001000	10	1168	1169	I - format
BL	100101	6	1184	1215	B - format
ORR	10101010000	11	1360		R - format
ADDS	10101011000	11	1368		R - format
ADDIS	1011000100	10	1416	1417	I - format
CBZ	10110100	8	1440	1447	CB - format
CBNZ	10110101	8	1448	1455	CB - format
STURW	10111000000	11	1472		D - format
LDURSW	10111000100	11	1476		D - format
STXR	11001000000	11	1600		D - format
LDXR	11001000010	11	1602		D - format
EOR	11101010000	11	1616		R - format
SUB	11001011000	11	1624		R - format
SUBI	1101000100	10	1672	1673	I - format
MOVZ	110100101	9	1684	1687	IM - format
LSR	11010011010	11	1690		R - format
LSL	11010011011	11	1691		R - format
BR	11010110000	11	1712		R - format
ANDS	11101010000	11	1872		R - format
SUBS	11101011000	11	1880		R - format
SUBIS	1111000100	10	1928	1929	I - format
ANDIS	1111001000	10	1936	1937	I - format
MOVK	111100101	9	1940	1943	IM - format
STUR	11111000000	11	1984		D - format
LDUR	11111000010	11	1986		D - format

II. Required design documentation

1. **Register file:** With appropriate detail, show the layout of the register file.
2. **ALU:** You need to show a detailed design of an ALU to support the following instructions:

Table 4: Instruction set for required design

Instruction	Operation	ALU function
LDUR	load register	add
STUR	store register	add
CBZ	Conditional branch on zero	pass input b
CBNZ	Conditional branch on not zero	pass input b
B	Unconditional branch	
R-type	add	add
	subtract	subtract
	AND	AND
	ORR	OR

3. **Data Path:** Show the data path of each class of instruction as well as the overall data path per Table 4.
   a. **Control:** Develop the control logic to support your instructions. Gate level design in not necessary. However, functional table needs be very detailed and clear, making it implementable in an FPGA.
4. **Pipelined Data Path:** Show the data path for the pipelined architecture.

Extra Credits (20 Points):
Note: Extra Points are only considered if the processor design is over 80% correct. Either Verilog code or a schematic capture of the ALU is needed. Documentation should include two simulation cases for each ALU operation.

Design Team:
Two students per team is allowed. Only one report submission is needed per team.

Required Documentation and Due Dates

- **May 9, 2023**
  Final report: Only electronic submission via Brightspace Turnitin is accepted.

  You should include the following:
  2. A cover page indicating title of the project, course name and number, date (semester and year), and the name of each group member.
  3. Table of contents.
  4. Introduction - describing the project for a practicing engineering.
  5. Design – Detailed design as outline in Section II is required. I suggest the following sections:
     a. Machine code for the instruction set.
     b. A complete documented design of the ALU. This should be a paper design.
     c. Detail design of the data-path for the processor. This should be a paper design.
     d. Function tables of control logic.
  6. Conclusion (problems encountered, lessons learned, etc.).

Project guidelines:
- Late project is not accepted. If your project is not complete by the due date, you should hand in the incomplete project for a partial credit.
- Final report should be professionally documented. You should use a word processor and a CAD tool to document your work. You may not use existing MIPS diagram from your textbook and add circuitry to it.
- Your report needs to be free of grammatical and spelling errors.
- Your project should reflect your own work. If unreasonable similarities are recognized between the turned in projects, they will receive failing grades.

Applicability for Design Folder:
If your design and report are of acceptable quality, I will sign and mark the report as being acceptable to be included in your graduation design folder. Please note that as part of your graduate requirements, you need at least five design projects in your design folder.

In case, your report does not initially qualify to be included in your design folder, you may address the shortcomings and resubmit it for reconsideration to be included in the design folder. Please note this will not change your grade for the project or the course.