INSTRUCTION SETS

I GDE DHARMA NUGRAHA

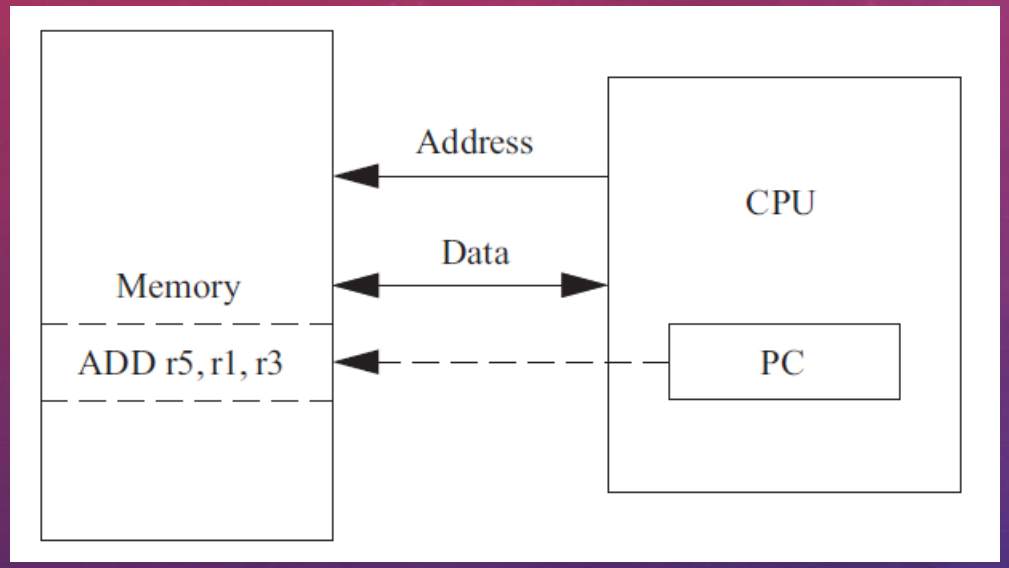
GOALS

- A brief review of computer architecture taxonomy and assembly language.
- Four very different architecture: ARM, PIC16F, TI C55x, and TI C64x

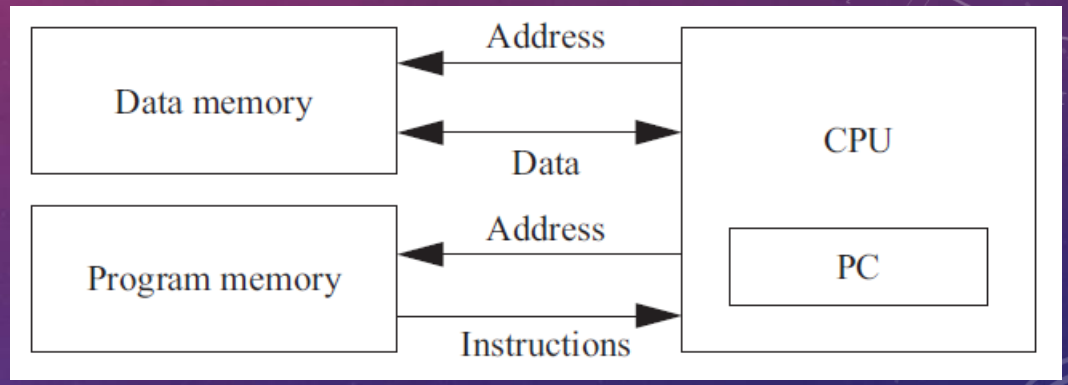
INTRODUCTION

- The instruction sets are the programmer's interface to the hardware.
- Understanding the instruction sets is the key to analyzing the performance of programs.
- By understanding the types of instructions that the CPU provides, the programmer will gain insight into alternative ways to implement a particular function.

COMPUTER ARCHITECTURE TAXONOMY



A von Neumann Architecture



A Havard Architecture

COMPUTER ARCHITECTURE TAXONOMY

- The computing system consists of a **Central Processing Unit (CPU)** and a **memory**.
- The memory holds both data and instruction and can be read or write when given an address.
- The CPU has several internal **register** that store values used internally.
  - One of those registers is the **program counter (PC)**, which holds the address in memory of an instruction.
- The CPU fetches the instruction from memory, decodes the instruction, and executes it.
- Data sets that arrive continuously and periodically are called **streaming data**.

COMPUTER ARCHITECTURE TAXONOMY

- Computer architecture relates to their instruction and how they are executed.
- The known instruction sets such as:
  - **Complex Instruction Set Computers** (CISC)
  - **Reduced Instruction Set Computers** (RISC)
    - Could be efficiently executed in **pipelined** processors.

COMPUTER ARCHITECTURE TAXONOMY

- Computers can be classified by several characteristics of their instruction sets.
- Instruction can have a variety of characteristics, including:
  - Fixed versus variable length;
  - Addressing modes;
  - Numbers of operands;
  - Types of operations supported.

COMPUTER ARCHITECTURE TAXONOMY

- Word length
  - The computer architecture can be characterized by their word length: 4-bit, 8-bit, 16-bit, 32-bit, and so on.
- The type of number bits, bytes and words
  - Little-endian mode → the lowest-order byte residing in the low-order bits of the word
  - Big-endian mode → the lowest-order byte stored in the highest bits of the word
- Instruction execution
  - Single-issue processor → executes one instruction at a time
  - Mutiple-issue processor → executes multiple instruction at a time
  - Superscalar processor → uses specialized logic to identify at run time instruction that can be executed simultaneously
  - Very Long Instruction Word (VLIW) processor → bergantung pada compiler untuk menentukan kombinasi instruksi yang bisa dieksekusi secara bersamaan.

COMPUTER ARCHITECTURE TAXONOMY

- The set of registers available for use by programs is called the **programming model**, also known as the **programmer model**.
- CPUs and systems
  - The CPU is only part of a complete computer system. In addition to the memory, I/O devices are needed to build a useful system.
  - A **microcontroller** is one form of a single-chip computer that includes a processor, memory and I/O devices.
  - A system on chip (SoC) generally refers to a larger processor that includes on-chip RAM that is usually supplemented by an off-chip memory.

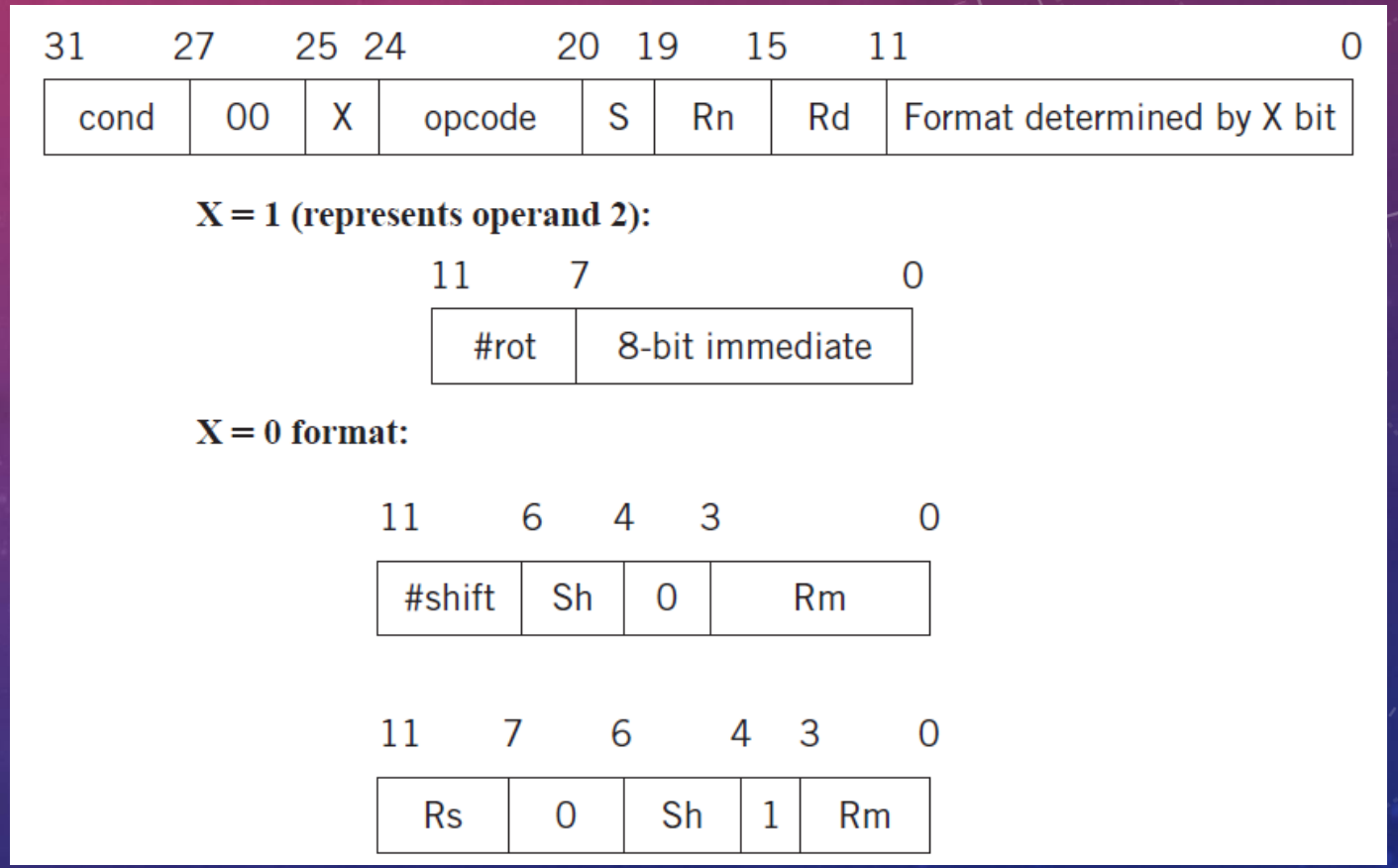
ASSEMBLY LANGUAGES

- Assembly Languages usually share the same basic feature:
  - One instruction appears per line;
  - **Labels**, which give names to memory locations, start in the first column;
  - Instructions must start in the second column of after to distinguish them from label;
  - Comments run from some designated comment character (; in the case of ARM) to the end of the line.

ASSEMBLY LANGUAGES

```
label1    ADR r4,c
          LDR r0,[r4]          ; a comment
          ADR r4,d
          LDR r1,[r4]
          SUB r0,r0,r1         ; another comment
```

An example of ARM assembly language



Format of an ARM data processing instruction

ASSEMBLY LANGUAGES

```
ADDGT r0, r3, #5
```

- the cond field would be set according to the GT condition (1100),
- the opcode field would be set to the binary code for the ADD instruction (0100),
- the first operand register Rn would be set to 3 to represent r3,
- the destination register Rd would be set to 0 for r0, and
- the operand 2 field would be set to the immediate value of 5.

ASSEMBLY LANGUAGES

- Assemblers must also provide some **pseudo-ops** to help programmers create complete assembly language programs.
  - An example: allows data values to be loaded into memory locations.

VERY LONG INSTRUCTIONS WORD (VLIW) PROCESSOR

- CPUs can execute programs faster if they can execute more than one instruction at a time.
- If the operands of one instruction depend on the results of a previous instruction, then the CPU cannot start the new instruction until the earlier instruction has finished.
- However, adjacent instructions may not directly depend on each other. In this case, the CPU can execute several simultaneously.
- One technique to address this problem is VLIW Processor.

VERY LONG INSTRUCTIONS WORD (VLIW) PROCESSOR

- VLIW Processor rely on the compiler to identify sets of instructions that can be executed in parallel.
- Superscalar processors can find parallelism that VLIW processors cannot -- some instructions may be independent in some situations and not others.
- However, superscalar processors are more expensive in both cost and energy consumption.
- Because it is relatively easy to extract parallelism from many DSP applications, the efficiency of VLIW processors can more easily be leveraged by digital signal processing software.

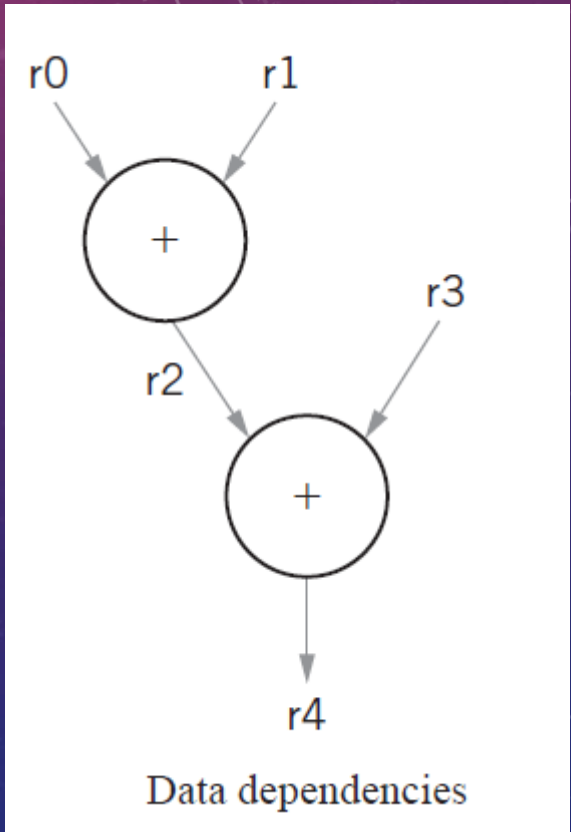
VLIW PROCESSOR

- In modern terminology, a set of instructions is bundled together into a **VLIW packet**, which is a set of instructions that may be executed together.
- The execution of the next packet will not start until all the instructions in the current packet have finished executing.
- The compiler identifies packets by analyzing the program to determine sets of instructions that can always execute together.

VLIW PROCESSOR

- The constraints for instructions to be executed in parallel:
  - A **data dependency**: a relationship between the data operated on by instructions.
  - Introduces **control dependencies**

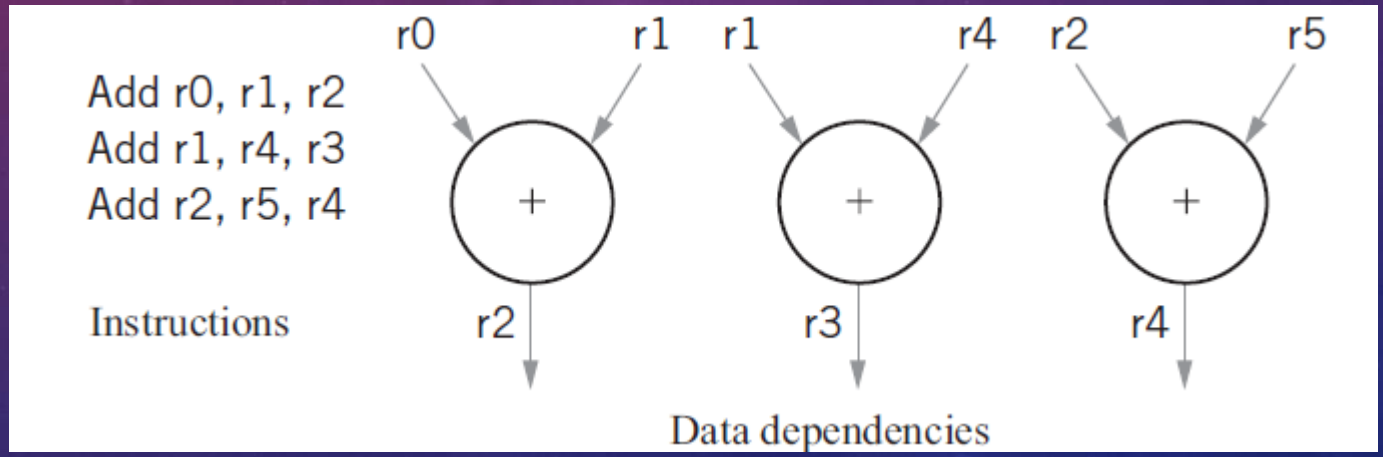
```
        bnz r3,foo
        add r0,r1,r2
foo:    ...
```



Data dependencies

VLIW PROCESSOR

- Opportunities for parallelism arise because many combinations of instructions do not introduce data or control dependencies.
- The natural grouping of assignments in the source code suggests some opportunities for parallelism that can also be influenced by how the object code uses registers.



VLIW PROCESSOR

- VLIW versus Superscalar
  - VLIW processors examine interinstruction dependencies only within a packet of instructions.
  - They rely on the compiler to determine the necessary dependencies and group instructions into a packet to avoid combinations of instructions that cannot be properly executed in a packet.
  - Superscalar processors, in contrast, use hardware to analyze the instruction stream and determine dependencies that need to be obeyed.

NEXT PROCESSOR

- ARM Processor
- PICmicro midrange family
- TI C55x DSP
- TI C64x

GROUP DISCUSSION

- Summary the processor and memory organization, addressing mode, data operations, flow of control and type of coding for each processor.

QUESTIONS?