Roadmap

Instruction Level Parallelism (ILP)

1. Review of Simple Pipelined Processors
2. Dynamically Scheduled Pipelines
3. Superscalar and Multithreaded Processors

1. Simple Pipelining

- Baseline design:
  - Introduce a simple RISC Instruction Set Architecture (ISA) (called DLX)
  - Non-pipelined DLX implementation

- 5-Stage Pipelined DLX Processor

- Hazards and their solutions

Roadmap

Instruction Level Parallelsim (ILP)

Overlapped execution of instructions from a sequential program

- Programmer obliviousness
- Single-threaded applications
- Compiler support

Limits

ILP Wall
Memory Wall
Power Wall

Simplified DLX Instruction Set

DLX: Idealized RISC processor (similar to MIPS, ARM)

- Load/Store architecture using base + offfset addressing
- 32 bit word size aligned at word address boundaries
- 32-bit memory addresses (aligned)

Registers

- 32 32-bit Integer GPRs R0 …. R31
- 32 32-bit Floating Point Registers F0, .. F31
- No condition codes

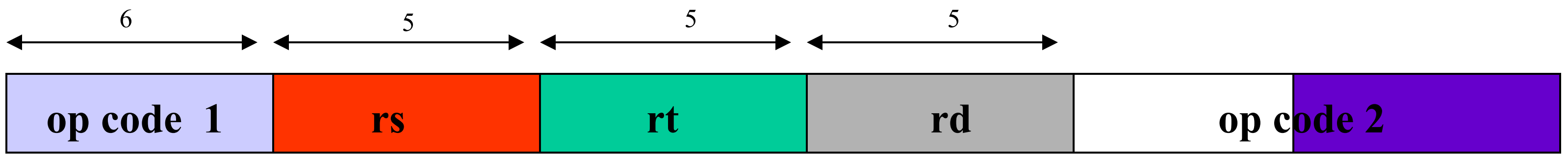
Instructions

- Integer ALU
- Floating Point (FP)
- Memory Access
- Program Control :

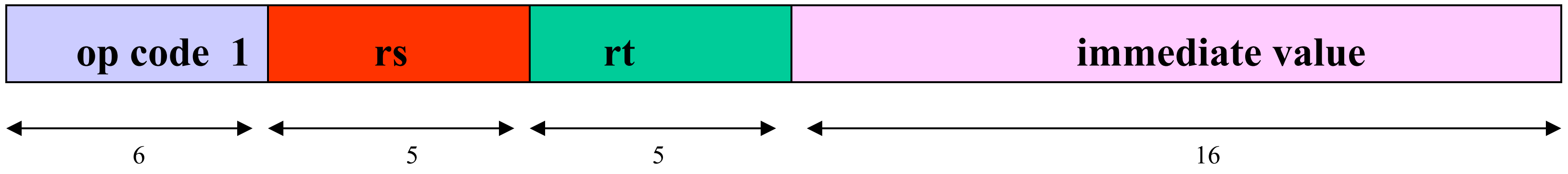
Instruction Formats

All instructions are one word long (32 bit aligned)

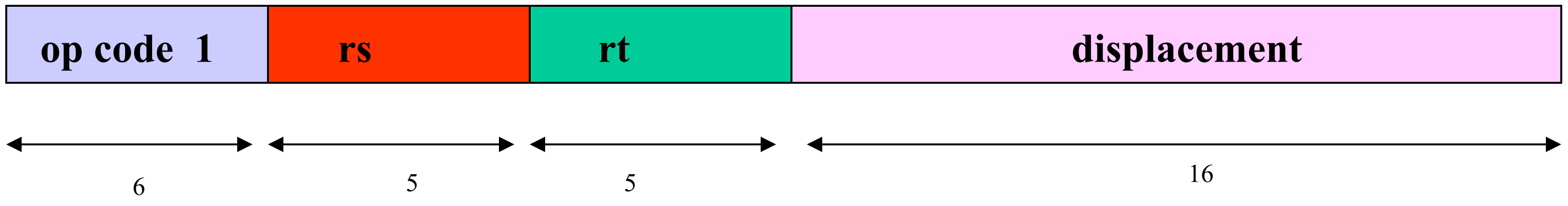
R-R Instructions: add rd, rs, rt



R-I Instructions: addi rt, rs, d



Load/Store/Branch Instructions: lw rt, d(rs), sw rt, d(rs), beq rs, rt, d(PC)



Simplified Integer DLX Instruction Set

1. ALU instructions: (ADD, SUB, AND, OR, XOR, ….)

RR mode: ADD rd, rs, rt | [rd] = [rs] + [rt]

Example: ADD R2, R4, R5

The 32-bit contents of registers rs and rt are added, and the sum is written to register rd.

RI mode: ADDI rt, rs, d | [rt] = [rs] + EXT(d)

Example: ADDI R2, R4, 1000

The 16-bit value d is sign-extended to 32 bits and added to the 32-bit contents of register rs. The sum is written to register rt.

Simplified Integer DLX Instruction Set

2. Memory reference instructions: Load (LW) and Store (SW)

```
LW  rt, d(rs)     | ea = EXT(d) + (rs);  (rt)  =  MEM[ea]
SW  rt, d(rs)     | ea = EXT(d) + (rs);  MEM[ea] =  (rt)
```

Example:

```
LW   R5, 0(R2)       | R5 gets the word whose address is stored in R2
SW  R6, 1000(R2)     | R5 written to memory location with address
                     | 1000  +  value in R2
```

- LW: Reads a word from memory at effective address [ea] and writes it to register rt
- SW: Writes the contents of register rt to memory at effective address [ea]

- Effective Address ea
  - 16-bit displacement d is sign-extended to 32 bits and added to the contents of base register rs to get the effective address.

Simplified Integer DLX Instruction Set

3. Control instructions

Conditional Branch

```
if condition is TRUE
        go to Target Address;
else
        continue with next in-line instruction
```

- Compare values in two registers (e.g. BEQ, BNE, BGT, BLT, BGW, BLE, …)
- Compare with 0 (e.g. BEQZ, BNEZ, BGTZ, BLTZ, BGEZ, BLEZ, …)

Example

```
BEQ     rs, rt, d(PC)   | Go to Target Address if contents of registers rs and rt are equal
BEQZ    rs, d(PC)       | Go to Target Address if contents of register rs equals zero
```

Target Address = PC + 4 + Extended(d)

16-bit displacement d is sign-extended to 32 bits

PC is address of the branch instruction