EECS 370 Discussion


You know when you're falling asleep, and you imagine yourself walking or something.

And suddenly you misstep, stumble, and jolt awake?

Yeah!

Well, that's what a segfault feels like. Double-check your damn pointers, okay?

xkcd.com
EECS 370 Discussion

Topics Today:

– Control Hazards

– Branch Prediction

– Project 3

– stackoverflow Example
EECS 370 Discussion

Control Hazards

Key Concept

Which LC-2K instruction(s) can cause a Control Hazard?

In which stage are branches resolved?
Control Hazards

Key Concept

Which LC-2K instruction(s) can cause a Control Hazard?
   BEQ & JALR

In which stage are branches resolved?
   MEM
EECS 370 Discussion

Control Hazards

Problem:
If we don’t know what the next PC should be what do we do?

Options:
No Branches
Avoid
Detect-and-stall
Speculate-and-squash
EECS 370 Discussion

Control Hazards

1) No Branches

Is this a feasible solution?

How could we eliminate this if statement?

```c
if (r0 == r1) {
    r2 = r3;
} else {
    r2++;
}
```
Control Hazards

1) No Branches

Is this a feasible solution?

How could we eliminate this if statement?

Conditional Assembly!

```assembly
if (r0 == r1) {
    r2 = r3;
} else {
    r2++;
}
```
2) Avoid

ADD 1 1 1
NAND 2 2 2
BEQ 3 0 jump
ADD 4 4 4
ADD 6 6 6
NAND 7 7 7
Control Hazards

3) Detect-and-stall

Any better than avoid?
EECS 370 Discussion

Control Hazards

4) Speculate-and-squash

Guess!

What do you have to do if you’re correct?

What do you have to do if you’re wrong?
EECS 370 Discussion

Control Hazards

4) Speculate-and-squash

Guess!

What do you have to do if you’re correct?
Nothing at all

What do you have to do if you’re wrong?
Turn IF, ID, & EX into Noops
EECS 370 Discussion

Branch Prediction

Based on the PC

What did this branch do last time?

1-bit predictor

Not Taken

Not Taken

Taken

Taken
EECS 370 Discussion

Branch Prediction

2-bit predictor

- Weakly Taken
- Strongly Taken
- Not Taken

Strongly Not Taken → Weakly Not Taken → Weakly Taken → Strongly Taken

Taken → Not Taken → Taken → Not Taken → Taken
Branch Prediction

What does the PC become if we predict Not Taken?

What does the PC become if we predict Taken?
EECS 370 Discussion

Branch Prediction

What does the PC become if we predict Not Taken?
PC+1

What does the PC become if we predict Taken?
???
EECS 370 Discussion

Branch Prediction

Branch Target Buffer

Maps PC values to Addresses

<table>
<thead>
<tr>
<th>PC</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x1234</td>
<td>0x00001000</td>
</tr>
<tr>
<td>0x4523</td>
<td>0x00004000</td>
</tr>
<tr>
<td>0xA342</td>
<td>0x00000004</td>
</tr>
<tr>
<td>0xFF76</td>
<td>0x0000A342</td>
</tr>
</tbody>
</table>
EECS 370 Discussion

Branch Prediction

<table>
<thead>
<tr>
<th>PC</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x1234</td>
<td>0x00001000</td>
</tr>
<tr>
<td>0x4523</td>
<td>0x00004000</td>
</tr>
<tr>
<td>0xA342</td>
<td>0x00000004</td>
</tr>
<tr>
<td>0xFF76</td>
<td>0x0000A342</td>
</tr>
</tbody>
</table>

Do we branch?

Where?
EECS 370 Discussion

Project 3

Make your own pipelined processor

• Need to use code for the file input from Project 1

• Write non-hazard code first

• Design good test code!!
EECS 370 Discussion

WBEND Register and Forwarding
Real-world example. Why this stuff matters.

public class Main
{
    public static void main(String[] args)
    {
        // Generate data
        int arraySize = 32768;
        int data[] = new int[arraySize];

        Random rnd = new Random(0);
        for (int c = 0; c < arraySize; ++c)
            data[c] = rnd.nextInt() % 256;

        // !!! With this, the next loop runs faster
        Arrays.sort(data);

        // Test
        long start = System.nanoTime();
        long sum = 0;

        for (int i = 0; i < 100000; ++i)
        {
            // Primary loop
            for (int c = 0; c < arraySize; ++c)
            {
                if (data[c] >= 128)
                    sum += data[c];
            }
        }

        System.out.println((System.nanoTime() - start) / 1000000000.0);
        System.out.println("sum = " + sum);
    }
}
public class Main
{
    public static void main(String[] args)
    {
        // Generate data
        int arraySize = 32768;
        int data[] = new int[arraySize];

        Random rnd = new Random(0);
        for (int c = 0; c < arraySize; ++c)
            data[c] = rnd.nextInt() % 256;

        // !!! With this, the next loop runs faster
        Arrays.sort(data);

        // Test
        long start = System.nanoTime();
        long sum = 0;

        for (int i = 0; i < 100000; ++i)
        {
            // Primary loop
            for (int c = 0; c < arraySize; ++c)
            {
                if (data[c] >= 128)
                    sum += data[c];
            }
        }

        System.out.println((System.nanoTime() - start) / 1000000000.0);
        System.out.println("sum = " + sum);
    }
}
int main()
{
    // Generate data
    const unsigned arraySize = 32768;
    int data[arraySize];

    for (unsigned c = 0; c < arraySize; ++c)
        data[c] = std::rand() % 256;

    // !!! With this, the next loop runs faster
    std::sort(data, data + arraySize);

    // Test
    clock_t start = clock();
    long long sum = 0;

    for (unsigned i = 0; i < 100000; ++i)
    {
        // Primary loop
        for (unsigned c = 0; c < arraySize; ++c)
        {
            if (data[c] >= 128)
                sum += data[c];
        }
    }

    double elapsedTime = static_cast<double>(clock() - start) / CLOCKS_PER_SEC;

    std::cout << elapsedTime << std::endl;
    std::cout << "sum = " << sum << std::endl;
}
int main()
{
    // Generate data
    const unsigned arraySize = 32768;
    int data[arraySize];

    for (unsigned c = 0; c < arraySize; ++c)
        data[c] = std::rand() % 256;

    // !!! With this, the next loop runs faster
    std::sort(data, data + arraySize);

    // Test
    clock_t start = clock();
    long long sum = 0;

    for (unsigned i = 0; i < 100000; ++i)
    {
        // Primary loop
        for (unsigned c = 0; c < arraySize; ++c)
        {
            if (data[c] >= 128)
                sum += data[c];
        }
    }

    double elapsedTime = static_cast<double>(clock() - start) / CLOCKS_PER_SEC;

    std::cout << elapsedTime << std::endl;
    std::cout << "sum = " << sum << std::endl;
}
```cpp
int main()
{
    // Generate data
    const unsigned arraySize = 32768;
    int data[arraySize];

    for (unsigned c = 0; c < arraySize; ++c)
        data[c] = std::rand() % 256;

    // !!! With this, the next loop runs faster
    std::sort(data, data + arraySize);

    // Test
    clock_t start = clock();
    long long sum = 0;

    for (unsigned i = 0; i < 100000; ++i)
    {
        // Primary loop
        for (unsigned c = 0; c < arraySize; ++c)
        {
            // Branch Prediction!
            if (data[c] >= 128)
                sum += data[c];
        }
    }

    double elapsedTime = static_cast<double>(clock() - start) / CLOCKS_PER_SEC;
    std::cout << elapsedTime << std::endl;
    std::cout << "sum = " << sum << std::endl;
}
```