Data Structures

Array addressing via index register

Array addressing via element pointer

Array Layout

Calculate $\sum_{k} x_k$

Data Structures

References for this chapter

Uwe R. Zimmer - The Australian National University
Data Structures

Calculating $\sum_{i=1}^{n} a[i]$

```c
int sum (unsigned int uints[], unsigned int from, unsigned int to) {
    unsigned int s = 0;
    for (unsigned int i = from; i <= to; i++) {
        s += uints[i];
    }
    return s;
}
```

Data Structures

Replacing multiplication with shifted index register

```c
int sum (unsigned int uints[], unsigned int from, unsigned int to) {
    unsigned int s = 0;
    unsigned int r4 = #4; // element size is 4 bytes
    unsigned int r5 = #0; // first_element_offset
    for (unsigned int i = from; i <= to; i++) { // i > from, i <= to
        s += uints[i];
        r5 += r4; // offset := offset + element_size
    }
    return s;
}
```

Data Structures

Replacing indices with offsets

```c
int sum (unsigned int uints[], unsigned int from, unsigned int to) {
    unsigned int s = 0;
    unsigned int r4 = #4; // element size is 4 bytes
    unsigned int r5 = #0; // first_element_offset
    for (unsigned int i = from; i <= to; i++) { // i > from, i <= to
        s += uints[i];
        r5 += r4; // offset := offset + element_size
    }
    return s;
}
```

Data Structures

Assembling non-empty arrays

```c
int sum (unsigned int uints[], unsigned int from, unsigned int to) {
    unsigned int s = 0;
    unsigned int r4 = #4; // element size is 4 bytes
    unsigned int r5 = #0; // first_element_offset
    for (unsigned int i = from; i <= to; i++) { // i > from, i <= to
        s += uints[i];
        r5 += r4; // offset := offset + element_size
    }
    return s;
}
```

Data Structures

Array Slices

```c
int sum (unsigned int uints[], unsigned int from, unsigned int to) {
    unsigned int s = 0;
    unsigned int r4 = #4; // element size is 4 bytes
    unsigned int r5 = #0; // first_element_offset
    for (unsigned int i = from; i <= to; i++) { // i > from, i <= to
        s += uints[i];
        r5 += r4; // offset := offset + element_size
    }
    return s;
}
```
Moving blocks of memory can be done even much faster with special hardware. DMA controllers...