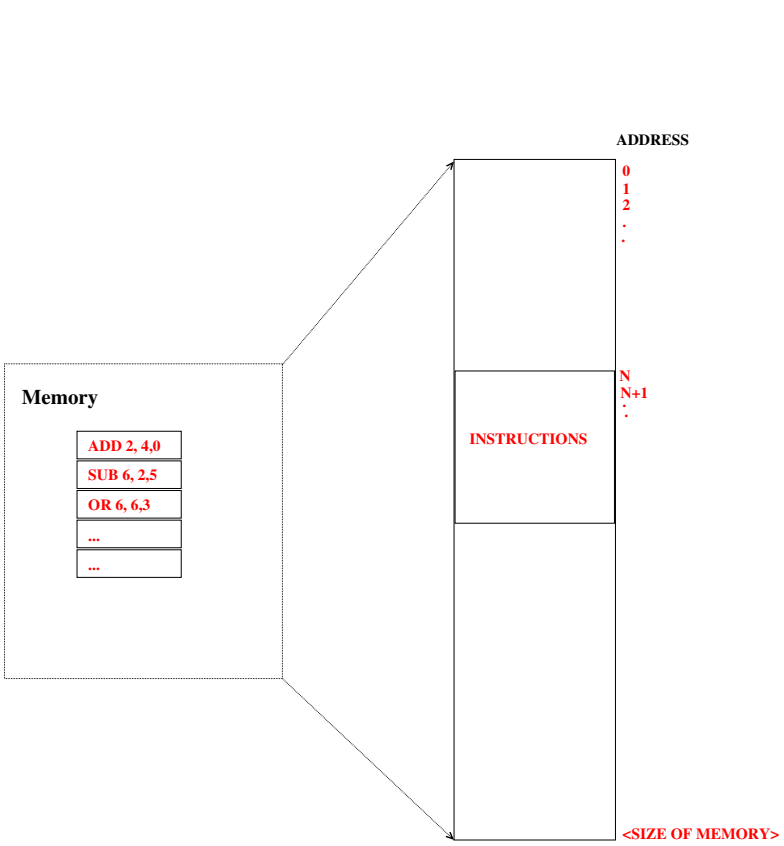
INF-1100
Pointers

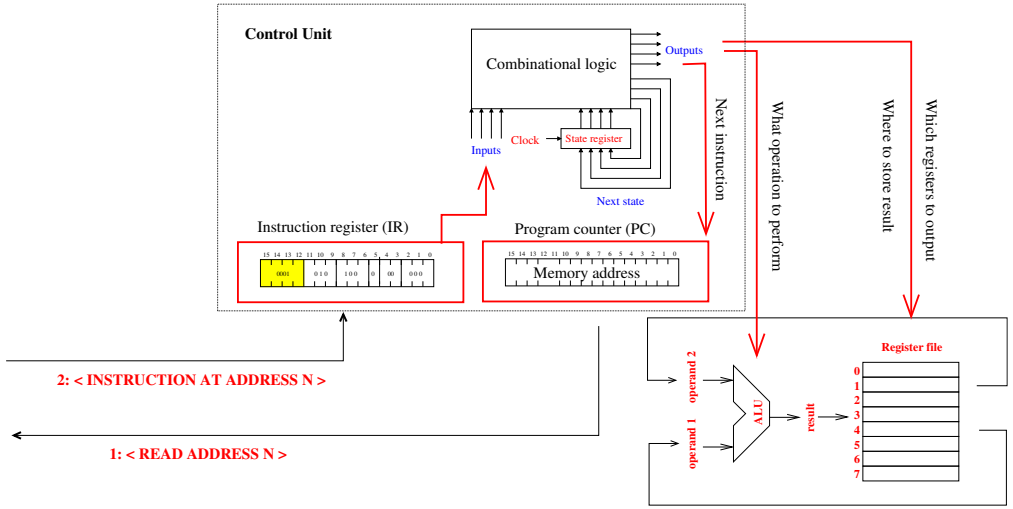
Åge Kvalnes

University of Tromsø, Norway

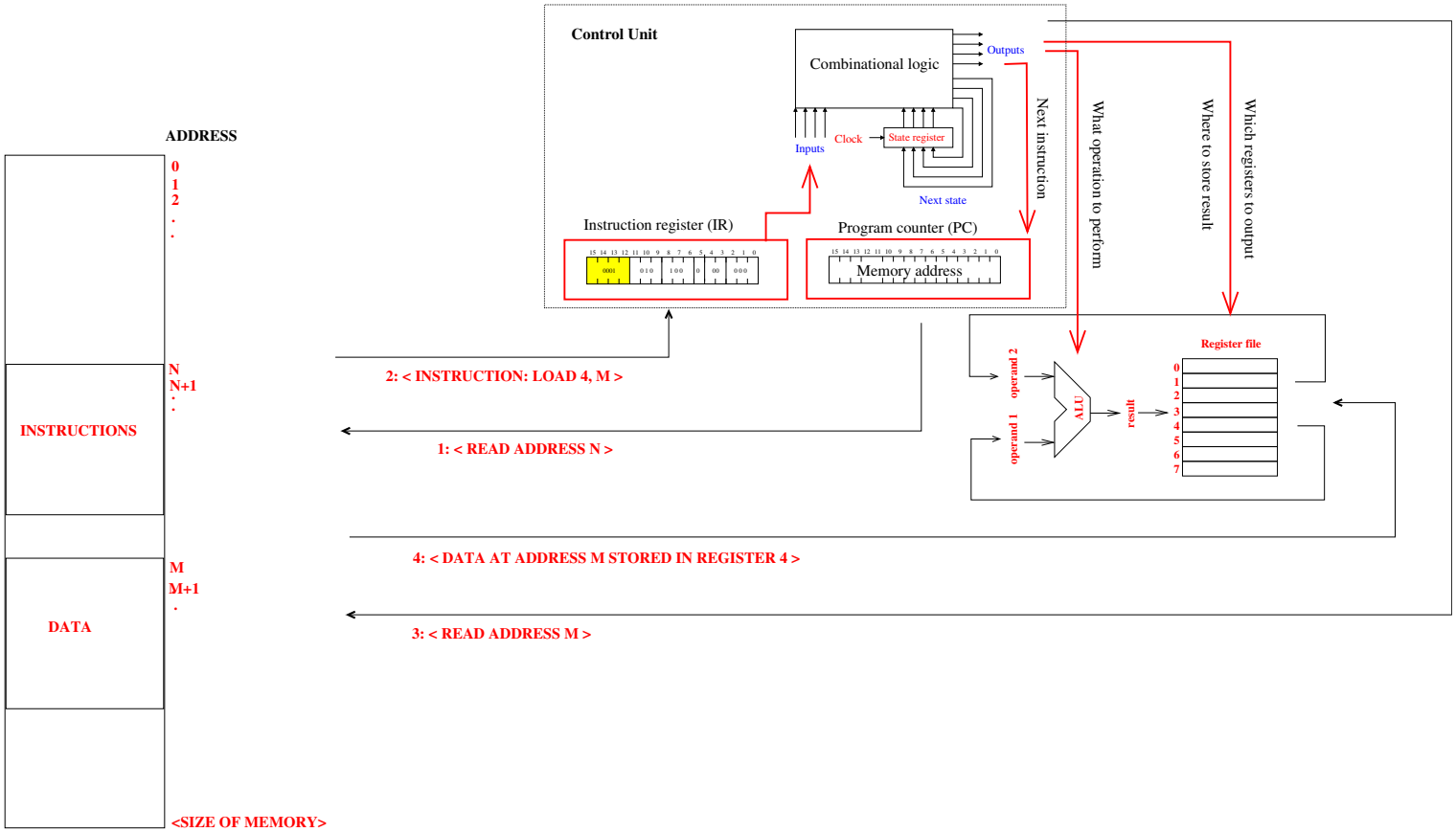
October 1, 2014

Instructions are located at a specific memory address



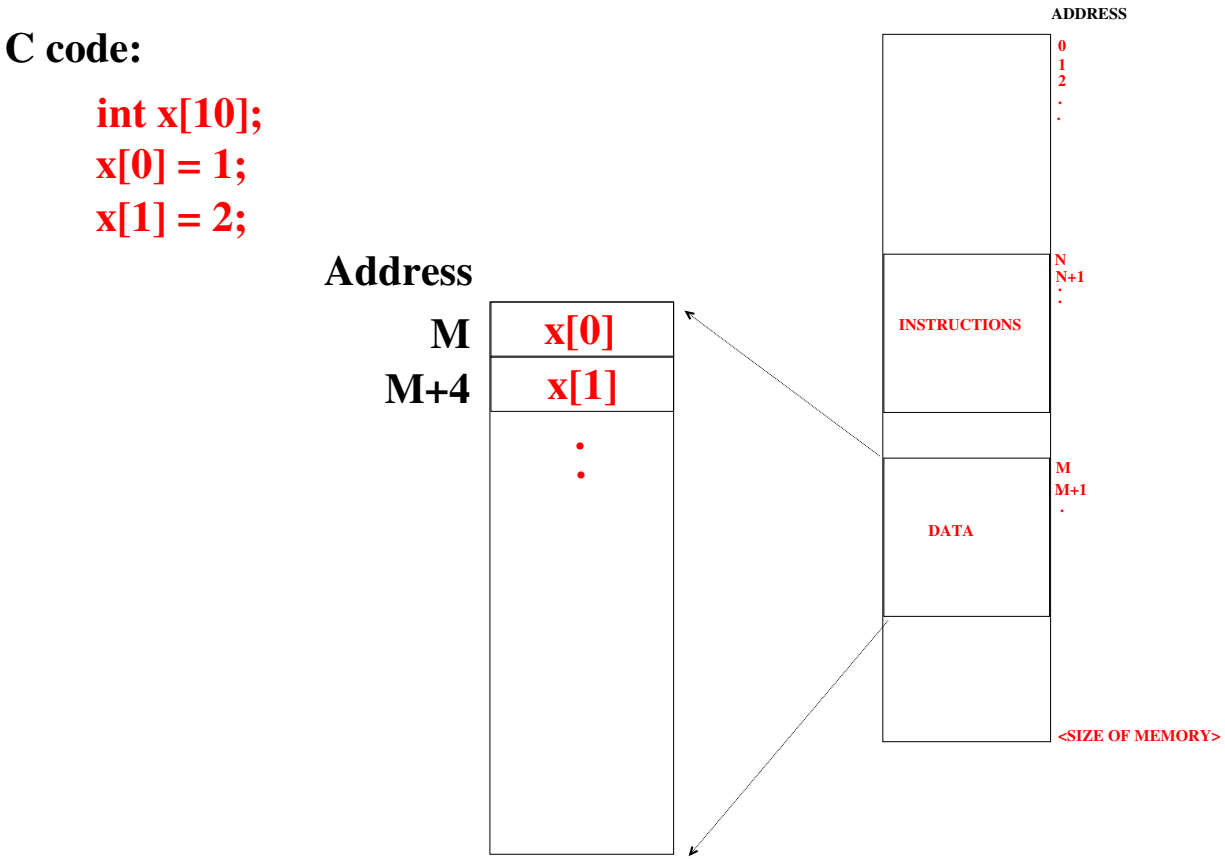


Data is located at a specific memory address



Data manipulated by instructions is also stored in memory

All variables are stored in memory at an address



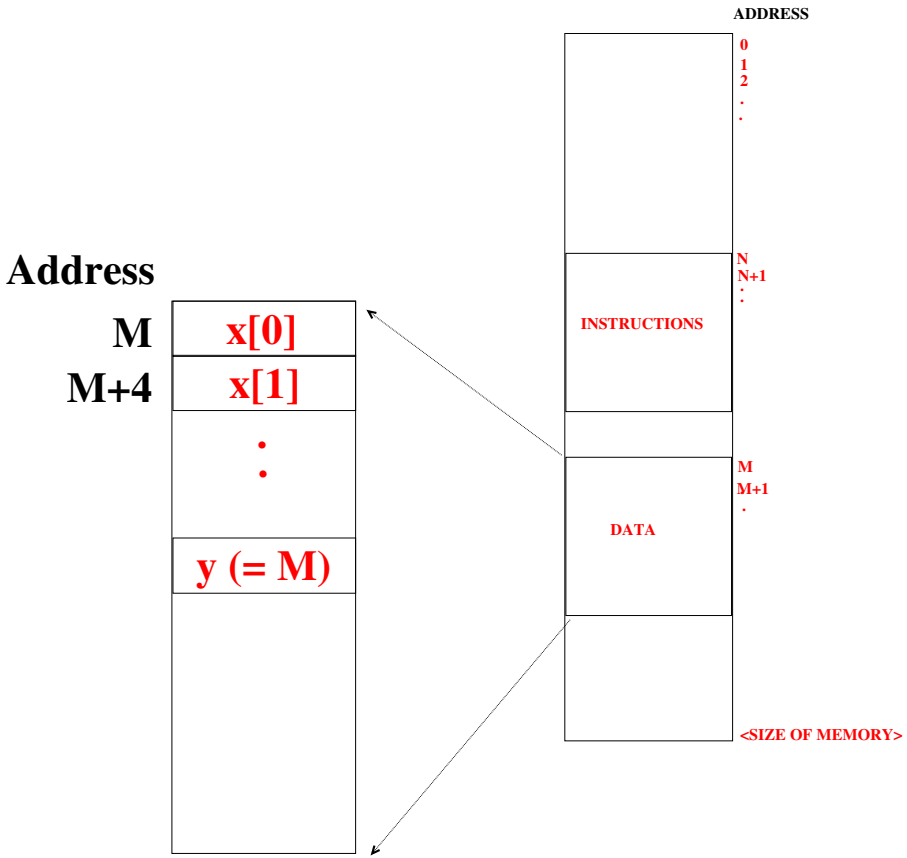
& in front of a variable equals the memory address of the variable

- &x[0] equals M
- &x[1] equals $M + 4$

A pointer is a variable containing a memory address

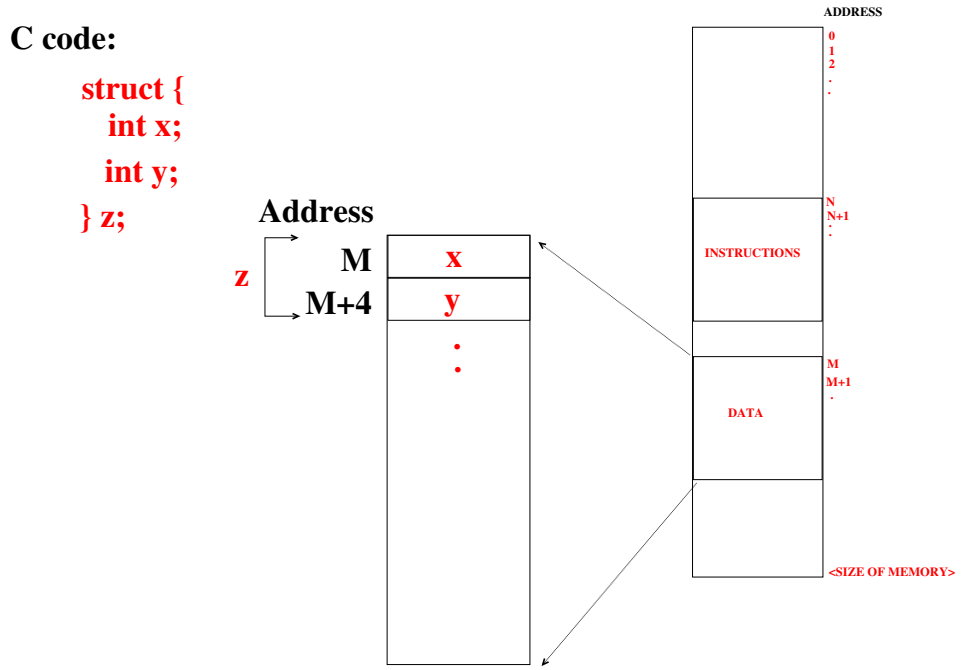
C code:

```
int x[10];
int *y;
x[0] = 1;
x[1] = 2;
y = &x[0]
```



Placing * after a type declaration creates a variable that will contain a memory address: a *pointer*.

Composite data structures



- z.x equals value of x
- z.y equals value of y
- &z.x equals M
- &z.y equals $M + 4$

Pointers to data structures

```
typedef struct mystruct mystruct_t;
struct mystruct {
  int number;
};

mystruct_t myvar;
mystruct_t *mypnt;

// Assign the memory address of myvar to mypnt
mypnt = &myvar;

// Assign value to myvar.number via mypnt
mypnt->number = 20;
```

Note use of $\rightarrow$ to access a field in myvar via mypnt.

Pointer arguments to functions

```
typedef struct mystruct mystruct_t;
struct mystruct {
  int number;
};

int myfunction(mystruct_t *mypnt)
{
  mypnt->number = 20;
}

int main(void)
{
  mystruct_t myvar;

  myfunction(&myvar);
}
```

Pointers and arrays

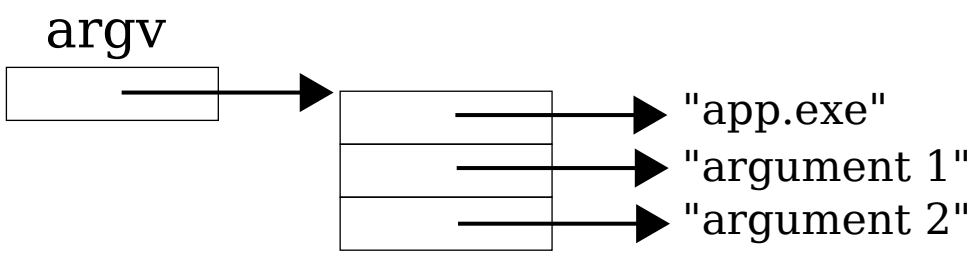
```
int myfunction(int *intpnt)
{
  intpnt[0] = 10;
  // Wrong, but possible.
  // Likely to cause the program to crash.
  intpnt[1] = 20;
}

int main(void)
{
  int intvar;
  myfunction(&intvar);
}
```

- Any pointer is assumed to point to the start of an array

The arguments to main

```
int main(int argc, char **argv)
{
}
```

- argv is a pointer to a pointer to a character.
- or argv is a pointer to an array containing pointers to text strings.
- argc specifies the length of the array.
- argv array contains pointers to command line arguments specified when starting the program.



Detecting circle overlap

Problem: Given a description of a set of circles (origin coordinate, radius), determine if some of the circles overlap.

Algorithm:

1. For each pair of circles, calculate orgin distance and check if less than the sum of radii.