Computer design

Example: Sum of array elements

P.J. Drongowski
SandSoftwareSound.net

C to hardware mapping example

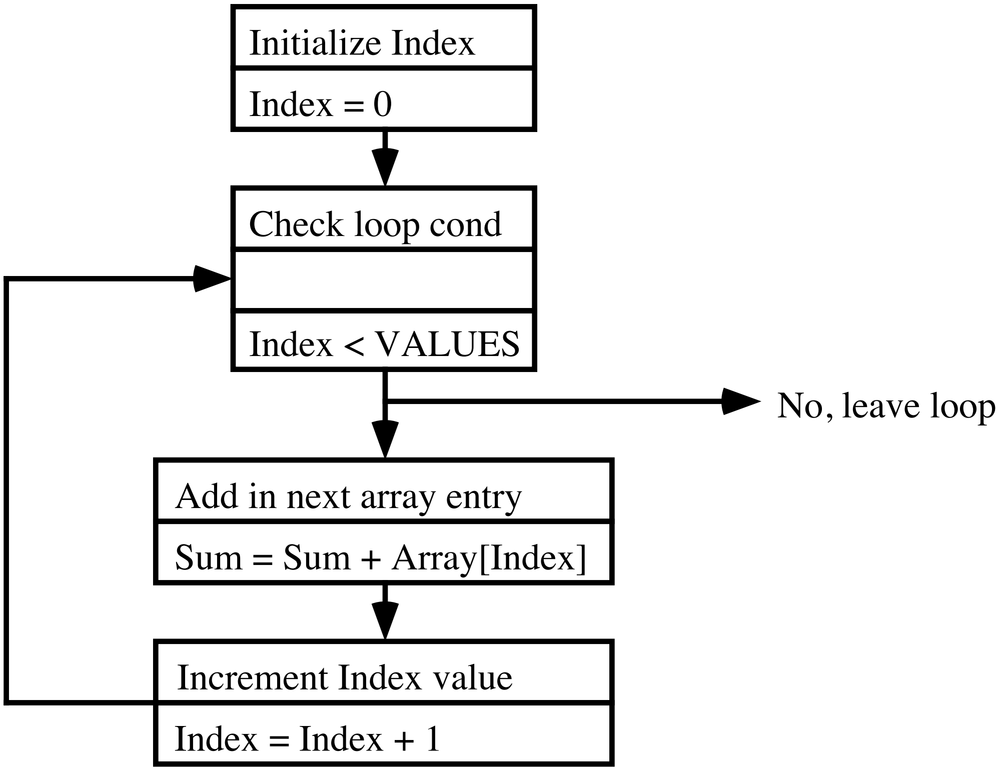
- Use simple translation rules
- Program to compute the sum of an array of values
- Version below has "internal" memory

```
#define VALUES 64

int Array[VALUES] ;

int Index,    /* Array index */
    Sum;      /* Current sum of values
*/

main()

  {
  Index = 0 ;

  while (Index < VALUES)
    {
    Sum = Sum + Array[Index] ;
    Index = Index + 1 ;
    }
```

- Mappings
  - Integer variable Index ⇒ up counter
  - Integer variable Sum ⇒ simple register
  - Map Array ⇒ 1-D RAM
  - Index = 0 ⇒ clear intrinsic operation
  - Index < VALUES ⇒ comparator
  - Sum = Sum + Array[Index] ⇒ adder
  - Index = Index + 1 ⇒ increment intrinsic

Sum over array example

```
#define VALUES 64

int Array[VALUES] ;

int Index,      /* Array index */
    Sum;        /* Current sum of values
*/

main()

  {
  Index = 0 ;

  while (Index < VALUES)
    {
    Sum = Sum + Array[Index] ;
    Index = Index + 1 ;
    }
```



Sum over array (example continued)

- Transfer-driven connections and expressions
  - Array[Index] ⇒ Index to Array
  - Sum = Sum + Array[Index]

    ⇒ Array to adder input

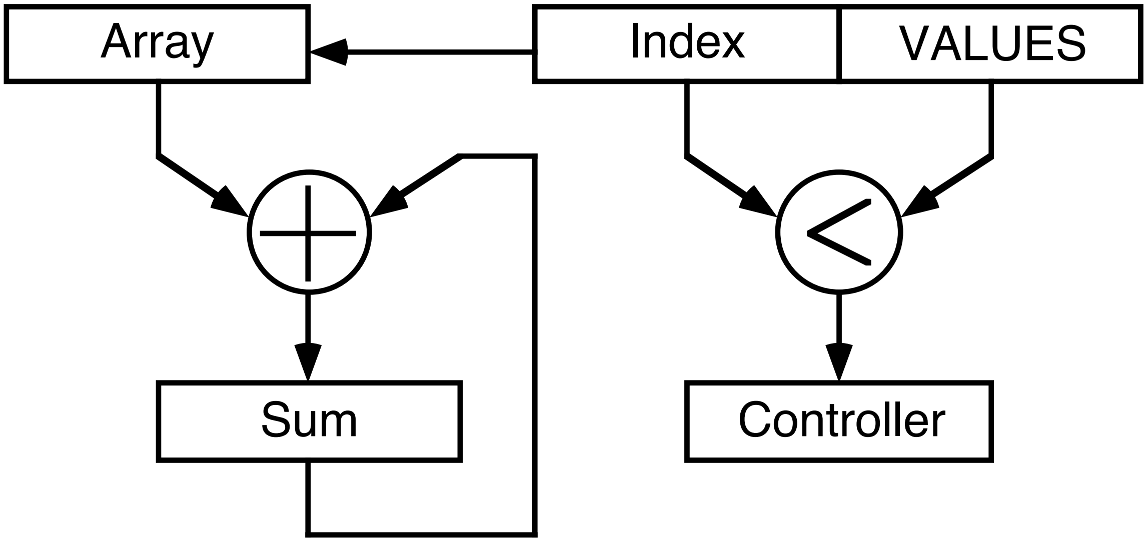
    ⇒ Sum to adder input

    ⇒ Adder output to input of Sum
  - Index < VALUES

    ⇒ Index to input of less than

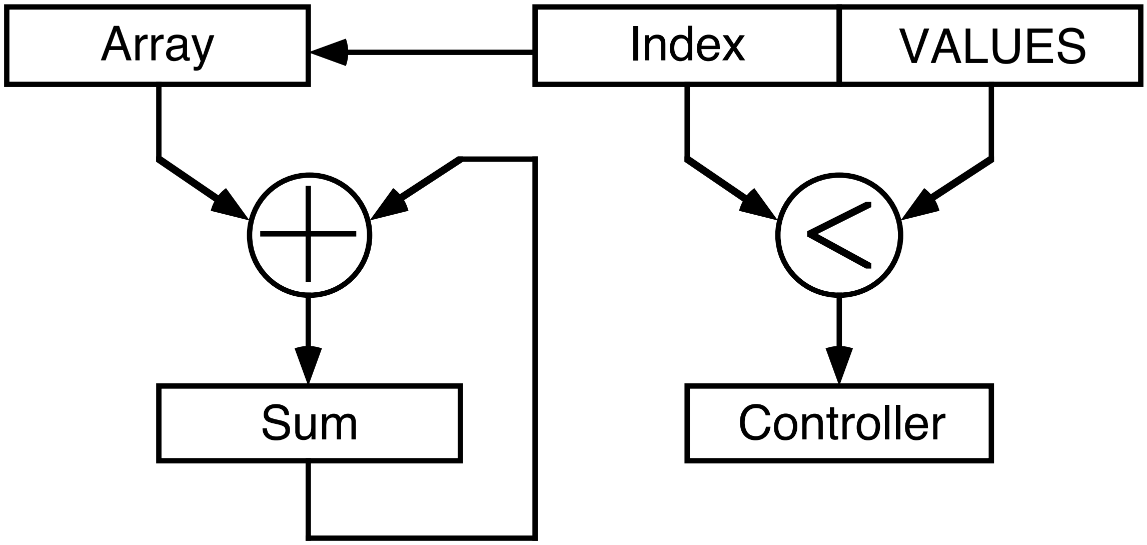
    ⇒ VALUES to input of less than

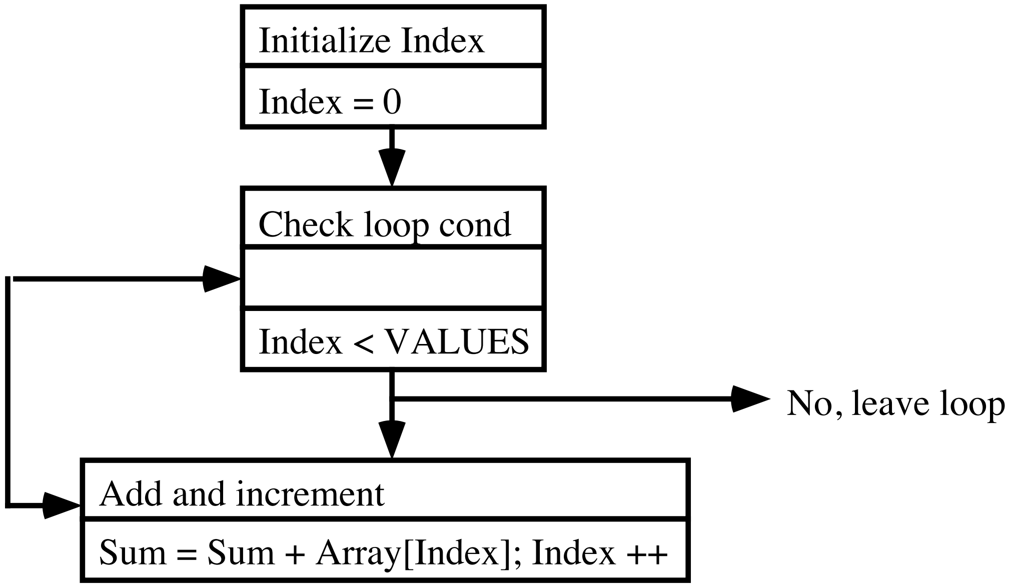
    ⇒ Output of less than to condition input of controller
- Observations
  - Sum must have master-slave operation
  - Sum and increment may be concurrent



Sum over array (parallel)

- Take advantage of datapath parallelism
- Perform sum and increment in same step





Initialize Index

Index = 0

Check loop cond

Index < VALUES

No, leave loop

Add and increment

Sum = Sum + Array[Index]; Index ++

Sum over array (clocking)

- How can we use Index to select an array element?
- And, do we use the old value or the new value of Index?

- C language assignment semantics
    - Example assignment

      Index = Index + 1

    - = is the assignment operator not equality!
    - The ← symbol would be a better choice¨

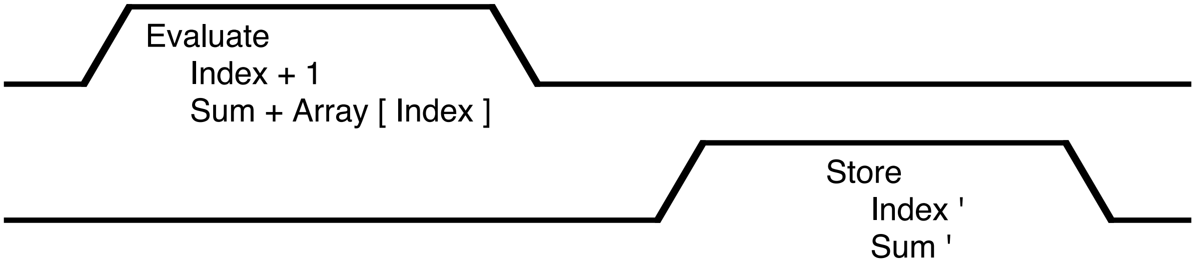
- This state - next state semantics
    - Formally specify state changes
    - "Prime" variables denote "the new value of"
    - Unprimed variables denote "the old value"
    - Examples

      Index' = Index + 1

      Sum' = Sum + Array [Index]

    - The = symbol denotes equality

- Two-phase, non-overlapping clock
    - Compute during Phi-1
    - Store during Phi-2



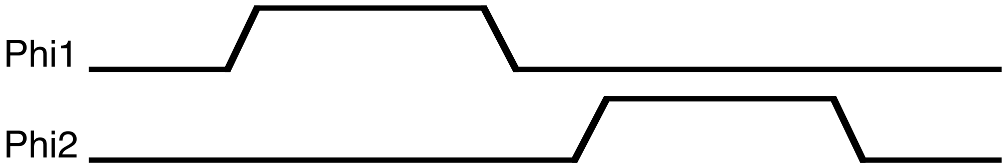
Evaluate
Index + 1
Sum + Array [Index]

Store
Index '
Sum '

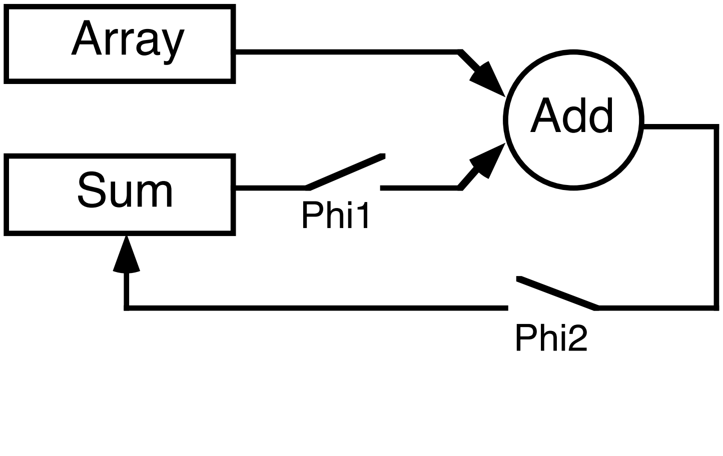
- Single assignment (a related concept)
    - Name appears to the left of ← at most once

Two phase clocking revisited

- Register "Sum" needs master-slave operation
- If D-type latches are used, output will follow input
- New value of Sum will feedback through D-type register
- Value will "race" around the loop



- Two-phase, non-overlapping clock gives us a solution
- Let the adder compute during Phi1
- Store the result on Phi2
- Use switches to pass
  - Old value of Sum to adder on Phi1
  - New value of Sum to the register on Phi2
- Scheme utilizes dynamic storage on input gates of adder


Array
Add
Sum
Phi1
Phi2